2017

A new framework and learning tool to enhance the usability of software

Almansour, Fahad

Copyright Statement

A new framework and learning tool to enhance the usability of software

By

Fahad Mohammed Almansour

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

Doctor of Philosophy

School of Computing, Electronics and Mathematics
Faculty of Science and Engineering

April 2016

Author's Declaration

At no time during the registration of the degree of Doctor of Philosophy has the author been registered for any other University award without the prior agreement of the Graduate Committee

Relevant scientific seminars and conferences were regularly attended at which work was presented and several papers were published.

Publications:

Almansour, F., & Stuart, L. (2014). Promoting the use of Design Evaluation Techniques within Software Development. In BCS HCI.

Almansour, F., & Stuart, L. (2016). Developing Innovative Tool to Enhance the Effectiveness of Decision Support System. ICDSST 2016, LNBIP 250, pp. 1–17.

Conferences Attended:

16th International Conference – Information Visualisation (IV 12),10/13 July 2012- Montpellier-France

26th Annual BCS HCI Conference (HCI 2012), 12/14 September 2012- Birmingham UK

27th Annual BCS HCI Conference (HCI 2013), 09/13 September 2013- London UK

28th Annual BCS HCI Conference (HCI 2014), 09/12 September 2014- Southport UK

Signed……. Dated: 12/04/2016

Thesis word count: 70,760

Abstract

Due to technological developments, apps (mobile applications) and web-based applications are now used daily by millions of people worldwide. Accordingly, such applications need to be usable by all groups of users, regardless of individual attributes. Thus, software usability measurement is fundamental metric that needs to be evaluated in order to assess software efficiency, effectiveness, learnability and user satisfaction. Consequently, a new approach is required that both educates software novice developers in software evaluation methods and promotes the use of usability evaluation methods to create usable products.

This research devised a development framework and learning tool in order to enhance overall awareness and assessment practice. Furthermore, the research also focuses on Usability Evaluation Methods (UEMs) with the objective of providing novice developers with support when making decisions pertaining to the use of learning resources. The proposed development framework and its associated learning resources is titled dEv (Design Evaluation), and it has been devised in order to address the three key challenges identified in the literature review and reinforce by the studies. These three challenges are: (i) the involvement of users in the initial phases of the development process, (ii) the mindset and perspectives of novice developers with regard to various issues as a result of their lack of UEMs or the provision of too many, and (iii) the general lack of knowledge and awareness concerning the importance and value of UEMs. The learning tool was created in line with investigation studies, feedback and novice developers requirements in the initial stages of the development process. An iterative experimental approach was adapted which incorporated the use of interviews and survey-based questionnaires. It was geared towards analysing the framework, learning tool and their various effects. Two subsequent studies were carried out in order to test the approach adopted and provide insight into its results. The studies also reported on their ability to affect novice developers using assessment methods and also to overcome a number of the difficulties associated with UEM application.

This suggested approach is valuable when considering two different contributions: primarily, the integration of software evaluation and software development in the dEv framework, which encourages professionals to evaluate across all phases of the development; secondly, it is able to enhance developer awareness and insight with regard to evaluation techniques and their application.

Acknowledgment

In the name of Allah, the Most Gracious and the Most Merciful Alhamdulillah, all praises to Allah for allowing me to complete this thesis.

I am heartily thankful to my supervisors, Dr Liz Stuart and Dr Guido Bugmann, who encouragement, guidance and support from the initial to the final stage enabled me to develop an understanding of the research process. Thanks also go to Prof Angelo Cangelosi for his valuable comments and suggestions during the transfer report viva. I would like to also thank staff and research students of the school of Computing, Electronics and Mathematics (Faculty of Science and Engineering) however my special thanks goes to Roy Tucker, Hussain Alsaiari, Torbjorn Dahl, Carole Watson and Julie Taylor.

I offer my kindest regards and blessings to all who supported me in any way during the completion of this thesis. These include University of Plymouth and the studies participants. I owe special thanks to my wonderful friends for their consistent support, encouragement and real friendship that I needed in Plymouth. Lastly, the completion of this thesis would not have been possible without the sponsorship and financial support of Al-Qassim University, Saudi Arabia.

Dedication

I dedicate this thesis to my Mother, Khadijah, and my father, Mohammed, and to the spirit of my grandmother, Turaifh, to whom I am indebted for the rest of my life.

I also dedicate this thesis to my lovely wife, Badria, for her patience and support; to my brothers, Khaled, Abdulrahman, Abdullah, Ahamed, Saad, Abdulelah and Abdulmajeed; and my sisters, Huda and Asma who encouraged me and who have taken care of my parents while I have been in the UK.

Publications

Almansour, F., & Stuart, L. (2014). Promoting the use of Design Evaluation Techniques within Software Development. In BCS HCI.

Almansour, F., & Stuart, L. (2016). Developing Innovative Tool to Enhance the Effectiveness of Decision Support System. ICDSST 2016, LNBIP 250, pp. 1–17. (Accepted)

Table of Contents

Contents

List of Tables

List of Figures

Glossary of Abbreviations

2D	Two-Dimensional space
3D	Three-Dimensional space
AEA	American Evaluation Association
ARIA	Accessible Rich Internet Applications
ASD	Agile software development
ASP.NET	Active Server Pages .NET
AUCDI	Agile and User Centred Design Integration
CSS	Cascading Style Sheets
dEv	Design Evaluation supporting tool
GOMS	Goals, Operators, Methods and Selection
GUI	Graphical User Interface
H0	Null Hypotheses
H1	Alternative Hypotheses
HC	Human Computer
HCI	Human Computer Interaction
HE	Heuristics Evaluation
HTML5	Hyper-Text Markup Language
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organisation for Standardisation
LGPL	Lesser General Public License
MMIs	Man-Machine Interfaces
OS	Operating System
Q_n	Question Number
ROI	Return on Investment
SDLC	System Development Life Cycle
SPSS	Statistical Package for the Social Sciences
SUS	System Usability Scale
TUI	Tangible User Interfaces

UCASD	User-Centred Agile Software Development
UCD	User Centred Design
UEMs	Usability Evaluation Methods
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WAI	Web Accessibility Initiative
WCAG	Web Content Accessibility Guidelines
XP	Extreme Programming

Chapter One: Introduction

1 Overview

When creating products, usability is the key objectivity, with usability recognised as fundamental to efficiency and throughput; therefore, this is critical to software development and business overall (Han, Yun, Kwahk & Hong, 2001). Software usability, as a concept, has been recognised by the International Organisation for Standardisation (ISO) as 'the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specific context of use' (Ramli & Jaafar, 2008). Nonetheless, usability has been wrongly categorised as an aspect of software development that may be linked to or otherwise added to the end of the cycle of development. This is commonly misunderstood as being one element of the product completion, which is entirely erroneous. Accordingly, novice developers need to take into account usability as fundamental to software success.

Design processes and evaluation methods are applied to create products that have an effect on the software usability level, thus meaning that there any lack of these two concepts will have an impact on the product's general usability. Assessment is regarded as being far more wide-ranging then functionality testing; therefore, a number of scholars recognised this as being concerned with what evaluators evaluate: (i) software quality, (ii) system usability, (iii) the degree to which users' requirements can be fulfilled, and (iv) the propensity to establish system issues. Also, it needs to be recognised that the latter is concerned with the system's user satisfaction (Dix, Finlay, Abwod & Beale, 2004; Stone, Woodroffe, Minocha & Jarrett, 2005). Notably, however, a number of novice developers choose not to provide assessment methods during the process of design or otherwise leave this until the end. A lack of assessment knowledge, combined with developer mindset, are recognised as the main factors behind the avoidance of assessment and evaluation by developers (Ardito, Buono & Caivano, 2014; Rosenbaum, Rohn & Humburg, 2000). The design framework adopted in order to create the product also could have an impact on the software's usability: for instance, more conventional methods, including the waterfall model, might not be valuable in some instances at it is not a linear and fixable method, meaning it is not able to progress from one phase in the design process to the next prior to begin completed. Nonetheless, in the modern-day world, the iterative approach is more and more commonly used for design creation, as in the case of agile approaches. Such iterative methods provide a greater degree of movement

flexibility between the stages of the development process; therefore, agile models are recognised as most suitable for integration with assessment techniques in an effort to enhance usability overall, with many integration frameworks having been established for such improvement, including the between agile and User Centred Design (UCD) integration.

This issue has received much attention by a number of scholars who are concerned with general usability, with a number of researchers, including Jakob Nielsen, Don Norman, Ben Shneiderman, Alan Dix and others, providing their own contributions towards usable software. Importantly, Nielson (1993) coined the term discount usability, which seeks to implement high-speed and low-cost assessment methods in consideration with evaluating and improving software usability. Furthermore, discount usability provides validation that inexpensive approaches—not only those that are costly—can affect software usability level (Nielsen, 1995b). Importantly, three different approaches are detailed as discount methods, including thinking aloud, scenarios and heuristics evaluation (HE) (Nielsen, 2009). Accordingly, the discount approaches are considered valuable for garnering quick results in the design of software, particularly for those developers who have little experience in the assessment of software.

1.1 Motivation of this Research

Researcher experience and the literature review are two critical elements driving the researcher in the completion of this work. Moreover, few works have examined the ability of the novice software engineer to carry out usability assessments for their products, which should be recognised as a motivational factor. As recognised by Ardito *et al*. (2014), novice behaviour may be affected by inadequate background knowledge, which subsequently affects an individual's perceptions and views. In an effort to circumvent this problem, novice developers are required to garner insight into and understanding of new behaviours that could facilitate greater knowledge. As a result of the lack of background knowledge held by various novice developers, they may be seen to behave as users (Roehm, Tiarks, Koschke, & Maalej, 2012). Such behaviour will mean some end user requirements and assessments are omitted. Furthermore, issues arise when novice developers choose to assess a product in line with what they believe is suited to them. Improving on this can be achieved through enhancing

overall background knowledge amongst such individuals in regards the issues of considering themselves as end users.

One of the key elements having a notable and direct impact on the usability evaluation performance completed by novice developers is self-efficacy. In this regard, self-efficacy may be defined as the beliefs of individuals pertaining to their overall ability to create designated levels of performance that influence events that subsequently affect their lives (Bandura, 1994). In line with the insight garnered in regards novice developers, it may be stated that there is a tendency amongst these individuals to create products that do not offer simplicity or ease of use; this could be owing to inadequate insight into the importance, procedures and definition of a usability evaluation process. Accordingly, such a lack of insight has a negative and detrimental effect on their self-efficacy; subsequently, this results in the view that the performance of such tasks is either problematic or altogether unnecessary. Such a negative view in this regard induces avoidance behaviours and poor performance in the evaluation proves. Accordingly, novice developers lack in the motivation and confidence to utilise such tools.

In an effort to overcome this issue, the solution presented offers a learning instrument with the ability to enhance the knowledge of novice developers in regards usability evaluation advantages and the overall importance attributed to producing a usable product. Furthermore, this tool teaches of the importance in performing evaluation tasks, which, in turn, improve confidence in the completion of tasks, thus resulting in positively influencing their self-efficacy in regards the use of an evaluation task in line with products (Pajares & Schunk, 2001). Accordingly, learning about assessment methods and techniques, and how usability evaluations can be carried out, are areas that should be promoted amongst new software engineers. Furthermore, a number of different evaluation methods have been detailed, some of which are recognised as costly whereas others are less expensive. Moreover, a number of assessment methods require a long period of time for completion, whilst others, in contrast, are much quicker. With this noted, the perspective of the quick and inexpensive evaluation method is another motivational factor encouraging the researcher to improve the overall usability of software through the application of quick and inexpensive approaches. The significant costs linked with usability evaluations carried out by specialists are regarded as an obstacle, and have the propensity to preventing software engineers from evaluating their products, which ultimately calls for methods that are less costly. Decreasing the time

necessary for completing assessments is also fundamental, meaning providing novice software developers with teaching about usability evaluation methods and adopting such methods throughout the development process has the ability to impact redesign time.

This study researcher believe that novice developers required for more evaluation practice in order to achieve product with less usability issues. Figure **1-1**shows the framework of the research thinking

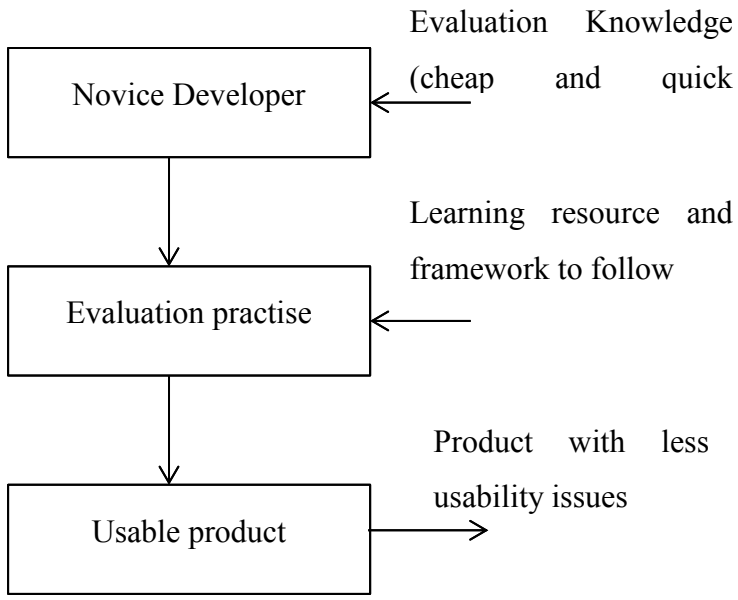


Figure 1-1: framework of the research thinking

1.2 Scope of the Research

This study centres on two key aspects. The first of these is the fact that novice software developer are regarded as key to the research, meaning there is a clear need for emphasis to be placed on their usability evaluation experience and their overall need to enhance their ability in completing usability evaluations and their focus on such. With this taken into account, investigation studies have been completed in order to establish the knowledge and requirements of the study subjects. Chapters 6, 8 and 9 provide a more in-depth explanation on this. The second key aspect in this work is user satisfaction, which is critical and emphasises establishing how evaluation strategies affect user satisfaction with products. In the present work, the system usability scale (SUS) and usability evaluation are carried out in mind of evaluating user satisfaction. The results of the usability evaluation for our suggested framework are detailed in Chapter 6. Furthermore, Chapter 9 details additional findings that provide further support for user satisfaction.

1.3 Research Questions

The following questions were formulated to act as a basis for this research:

1. What is the current status of usability evaluation practice for software engineering?
    a. What is the level of understating the evaluation methods?
    b. How does the current developer's knowledge of evaluation methods impact their practice?
    c. What is the relationship between developer's evaluation knowledge and the experience of software programming?
2. What steps should be taken by both software organisations and universities to promote the novice developers acceptance of evaluation methods in the software development process?
3. How can the learning resources help novice developers to increase the acceptance of the chosen evaluation methods in development process?

1.4 Aim and Objectives of the Research

The principle purpose of this research is to investigate, at the software development stages, the integration of evaluation methods required to create usable products. To achieve this purpose, seven unified objectives needed to be achieved in order to fulfil the abovementioned aim. These objectives are as follows:

Objective 1: To conduct a comprehensive literature review related to software usability, Usability Evaluation Methods (UEMs) and Human Computer Interaction (HCI) concepts.

Objective 2: To identify the current software developer's knowledge about usability evaluation methods (UEMs).

Objective 3: To identify the current developer's practice of conducting usability evaluation sessions.

Objective 4: To develop a theoretical model of the development process based on objectives 2 and 3, to promote the integration of evaluation methods in development process.

Objective 5: To assess how the new integration model impacts the developer's compliance for building usable products.

Objective 6: To assess how the use of new anticipated model impacts user satisfaction with products created based on it.

Objective 7: To draw, from the lessons learned, a set of recommendations to help in educating novice developers in applying usability evaluation methods by themselves towards

usable product; to point out the limitations of this work and indicate directions for future research.

1.5 The Significance of the Research

Through the completion of the initial literature review, software evaluation was recognised as the key phase in establishing usable software. Moreover, the key difference in gaining insight into the usability perspective and evaluation methods affect the overall process of designing usable software. Moreover, ensuring the evaluation methods are integrated within the software development process is valuable in enhancing the overall usability of the software, meaning that both practitioners and academics who show enthusiasm on the on-going application of such integration models are continuing to lack in knowledge and understanding surrounding the concept of software evaluation. With this taken into account, the present work has a number of valuable implications not only for practitioners but also for academics. Notably, from an academic standpoint, this study provides a learning resource and design model that is geared towards enhancing general understanding pertaining to usability meaning and usability assessment techniques. It has been established already that learning resources are critical in ensuring usability can be enhanced (Bruun & Stage, 2014; Howarth, Smith-Jackson, & Hartson, 2009; Skov & Stage, 2005).

The term Pedagogy has been defined by the Cambridge Dictionary as the examination of teaching-related activities and methods (dictionary.cambridge.org, 2017). Importantly, the word Pedagogy comes from the Greek words ‘peda’, meaning ‘child’, and ‘agogos’, meaning ‘study’. Importantly, the overall concept of Pedagogy maintains that teachers are responsible for the learning process, which encompasses assessments, content, presentations and structure (Tomei, 2010). In this vein, it has been noted by authors that classic pedagogy needs to incorporate various key principles in order to be a well aligned teaching tool. Primarily, one of the principles is ‘make haste slowly’, is recognised as valuable for learners in advocating the investment of the right amount of time to learn and complete assessments prior to moving on to the next stage (Glover & Miller, 2003). This particular aspect has been satisfied in the learning resource by allowing novice developers to take adequate time and accordingly learn and apply their own understanding to the study experiments. Secondarily, the concept of *multum non multa* infers the need of learners to learn a lot but not an excessive amount, with learners needing to be afforded with a few new teachings in-depth as opposed to many more

things on a smaller scale (Campbell, 2013). In this particular work, learning resources provide quick and simple evaluation methods, the aim of which is required learning. Accordingly, there are only a select few topics presented with in-depth information pertaining to each of the topics so as to ensure learner understanding. One further principle inherent in Pedagogy is that of repetition, which is recognised as the repetition of the mother of the learning (Gathercole, 1995). Accordingly, the study learning resource outlined various stages necessitating adherence in order to achieve task completion. Moreover, these individual stages may be repeated on two or more sections, meaning the repetition of the learning resource allows the learners to garner more in-depth understanding, with knowledge ingrained, as opposed to learners needing to make regular reference to the topic. Such a method is concerned with repeating knowledge at different stages so as to enhance the learning of novice learners.

Owing to the recognition of technology as fundamental in day-to-day life, the link between education and Pedagogy has become fundamental. Accordingly, various models have been devised and implemented in an effort to provide a strong basis and justification for such integration. For instance, a framework was designed by Mishra & Koehler (2006), which comprised technological, pedagogical and content knowledge (TPACK), with the model leading authors to establish Technological Pedagogical Content Knowledge-Web (TPCK-W), the latter of which sought to determine the more advanced knowledge required when teaching specifically on the web (Koehler, Mishra, Kereluik, & Shin, 2014). Accordingly, an online resource may be viewed as a pedagogical means of communicating knowledge; in this vein, it is stated by Ward &Benson (2010) that the approach selected is pivotal to achieving success in the online learning domain (Ward & Benson, 2010). Furthermore, In line with the researcher's own teaching background, coupled with the fact that online learning is viewed as one of the most valuable and effective methods of communicating knowledge to a large audience through the use of technology, as highlighted and supported by various scholars in the field (Carliner, 2004; Conrad, 2002) , online learning has been chosen to present our solution. This decision is further supported when considering the fact that it is a cost-efficient approach to teaching, and also is recognised as helping learners to prepare for a knowledge-based community, as noted by other academics in the arena (Appana, 2008; Katz, 1999). Therefore, such a learning resource has been designed in line with user requirements from those experiencing challenges in the field of usability evaluation. In addition, some of these obstacles will be discussed in terms of their proposed solutions. From a practitioner's

perspective, the study outcomes encourage practitioners to complete product usability assessment whilst also enhancing decisions made in the field of software usability.

1.6 Layout of the Thesis

This work is broken down into three key parts, beginning with an overview of the background, with framework production detailed in the second part and the contribution of the study in the final part. The thesis comprises nine individual chapters, as detailed in Table 1-2 . This structure has been adopted in mind of providing the reader with greater understanding of each chapter, on an individual basis, without any need to reference other parts of the study. Accordingly, the thesis layout is applied as follows:

- **Chapter 1: Introduction:** This chapter is concerned with highlighting the various issues, motivation and needs that are pertinent to the study, with subsequent efforts directed towards developing the research questions, aims and objectives. All of these are necessary to as to ensure the research process is guided.
- **Chapter 2: An Overview of Human Computer Interaction (HCI)—Background Theory:** This chapter provided a brief introduction of HCI and the meaning of it. HCI term is combination of three subjects: human behaviour, interface design and the interaction between humans and interfaces. Each subject is described in more detail. The impact of these factors on the production of practical ‘usable’ designs is assessed.
- **Chapter 3: An Overview of Software Development Methodologies—Background Theory:** This chapter reviews a number of software development methodologies that are applied to create software products. The Waterfall, Spiral and Agile software development methodologies are used in different projects; however, the agile approach is used widely at this time. The development techniques of User Centred Design (UCD), Scrum and extreme programming (XP) are examined. These development methodologies aim at improving the quality and usability of the software products.
- **Chapter 4: An Overview of Software Usability—Background Theory:** This chapter provides the concept of software usability and how to design usable products that are easy to use, before presenting a view of measurement elements and methods employed to assess the usability of the product. Interpreting the assessment results, it is important to understand the level of usability. Finally, the chapter is summed-up with ‘discount usability’ term and how it impacts the software usability.

- **Chapter 5: Integrating Agile with UCD towards A Theory of Integrated Development:** This chapter provides a practice of integration Agile and UCD. It is important to determine the integration ability, proposed principles, the integration benefits and challenges. Finally, the chapter is summed-up with our suggested development approach.
- **Chapter 6: Research Methodology:** This chapter provides a research philosophy, a discussion on the various data collection and analysis methods applied in the present work. Mix methods including number of data collection methods were carried out in a bid to garner empirical data. Convenience sampling has chosen in purpose to determine the study participants.

- **Chapter 7: Novice developers Investigation and Specification of New Tool Production:** This chapter provides the first investigation study into novice developers' evaluation knowledge, conducted to measure the knowledge of software engineers. The results of the investigation study led to meeting novice developers and collecting their requirements for new evaluation learning resources (study 2). This was followed by an evaluation study measuring learning resources before their application.
- **Chapter** 8**:** This chapter has detailed the creation of the dEv framework and how the various dEv integration methods are associated with the framework stage. The framework's challenges and impacts are also discussed in this chapter.
- **Chapter 9: Empirical Evaluation of dEv Tool Effects on Novice developers' Designing:** This chapter details an experiment concerned with the empirical evaluation of the dEv model and the effects of the learning resource from the perspective of design novice developers. Two groups performed the Shneiderman and dEv design frameworks, whilst the third group focused on the developer framework. The comparison work was completed on the basis of these three groups, concerned with establishing the number of involved users, the amount of acquired behaviour from all individual design frameworks, and the evaluation methods applied. The learning resource and dEv model were concerned with enhancing the perspective of novice developers in involving methods and users throughout the development process, centred on achieving a usable solution.

- **Chapter 10: Empirical Evaluation of dEv Tool Impacts on End-User Satisfaction:** This chapter provides an experiment study applying the dEv model as a supporting resource for novice developers. The study novice developers were divided into two

groups: the first was given the dEv learning resource as a support for software evaluation; the second group was not given any support. The final applications were evaluated and ranked according to the System Usability Scale (SUS). The results showed the dEv learning resource strongly impacts the usability of the final applications. Furthermore, research found that undergraduate software engineers are willing to conduct some of the evaluation methods under expert supervision.

- **Chapter 11: Conclusion—Contribution:** The principle aims of this chapter are to present a set of recommendations to promote the using of usability evaluation methods during the development process, and also to explain how this research contributes to the suability perspective. This chapter also presents some of the limitations of this study and suggestions for further research.

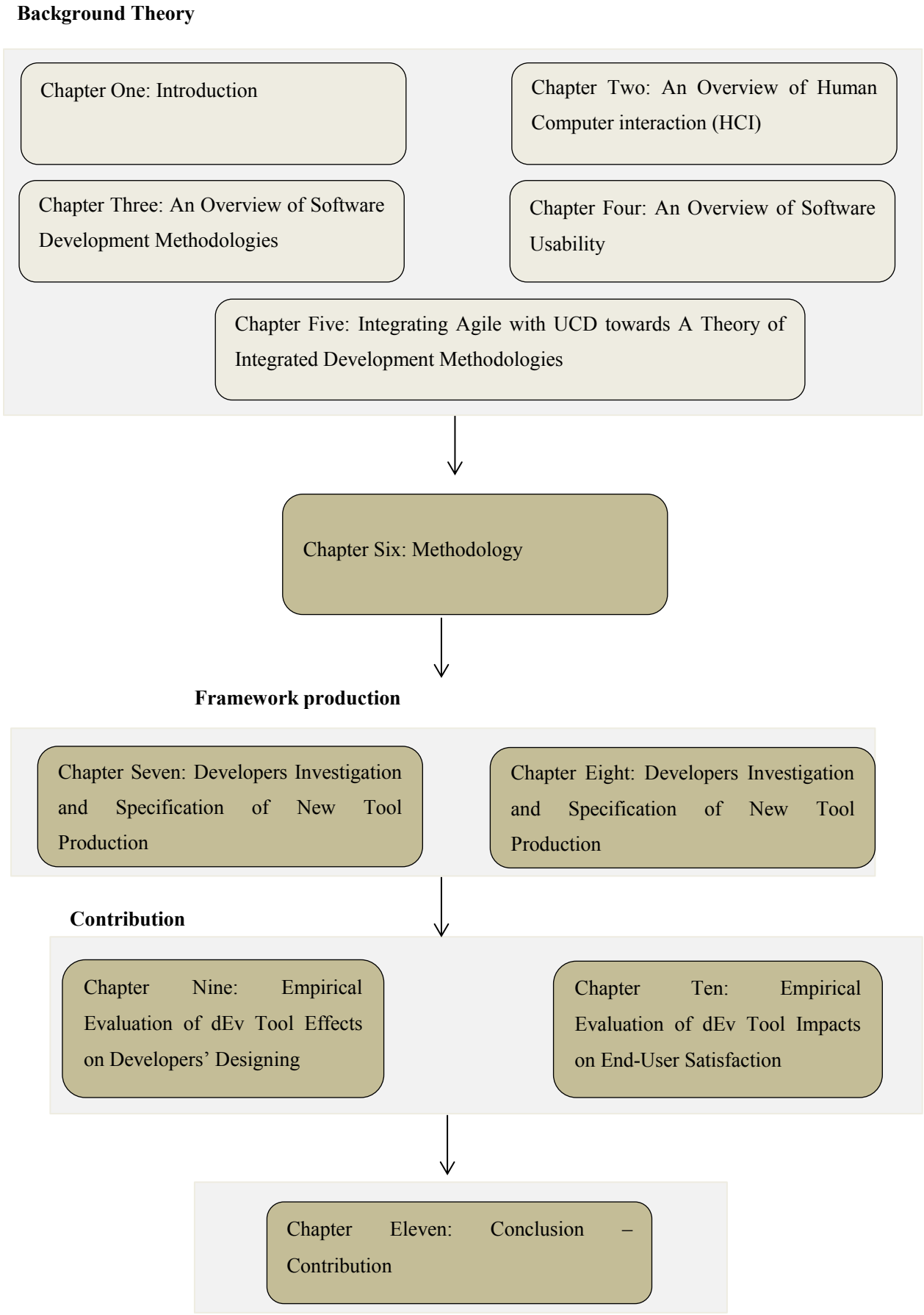


Figure 1-2: The structure of the thesis layout

Chapter Two: Human Computer Interaction

This chapter provides a brief introduction of HCI and the meaning of it. HCI is made up of a combination of three subjects: human behaviour, interface design and the interaction between humans and interfaces. Each of these subjects are described in detail. The impact of these factors on the production of practical “usable” designs is assessed.

2 Human Computer Interaction (HCI)

Human computer interaction (HCI) is “the study of human interaction with computer interfaces and the development of computer based interfaces to support effective user-computer tasks and interaction" (Debnath, Hussain, Islam, Hossain, & Rahman, 2012). This definition relates to three aspects: the first related to humans: the users who will use the product; the second related to computers: the technology which is used; and the third related to interaction, Interaction is defended as "the coordination of information exchange between the user and the system" (Juristo, Moreno, & Sanchez-Segura, 2007). This also describes the extent to which the system interacts with the user.

The subject of human computer interaction has existed as long as computers themselves. Initially, a foundation for the topic evolved during the 1970s when interest focused on Man-Machine Interfaces (MMIs). However, during the mid-1980s, the topic of how humans interact with computers became a major area of research interest(Carroll, 2014). At this stage, fundamental HCI techniques were used to improve MMIs which are described as ‘an input language for the user, an output language for the machine, and a protocol for interaction (Preece et al., 1994). In late 1990s, there was a substantial period of improvement when HCI was described as a fascinating area of research related to design, interaction and computers (Churchill, Bowser, & Preece, 2013; Shackel, 2009a).

2.1 Human Factors

A wide range of technology has emerged in our world. Humans interact with this technology in many different ways. The classical means of interaction: the keyboard and mouse is being augmented with touch screen interfaces and voice activated systems. The selection and design of an appropriate interface needs to consider the typical users’ needs when interacting with the system as well as the typical human’s abilities and expectations. One of the most important factors a system designer should consider is the typical human’s memory and cognition levels; in order to create design which is closely to the natural of human properties(Sutcliffe, 1988)

2.1.1 Memory and Cognition

The capacity of the human memory is one of the important concepts to consider when designing a user interface. In order to do this, an understanding of human memory is essential. There are three types of human memory: sensory memory, short-term memory (working memory) , and long-term memory (Benyon, 2010; Human-memory.net, 2010).

In sensory memory, a huge amount of information is held subconsciously. Auditory, visual, and tactile information fits into this category. Design should always aim to avoid unnecessary information, gregarious use of complex colour and icon as these will simply overload sensory memory. Such overloading will only serve to prevent the user from memorising the relevant / necessary information presented by the interface.

The second type of memory, Short-Term Memory, receives information from sensory memory and long term memory. Thus, short term memory has a limited capacity to recall items. Indeed, Mandel (1997) quantifies that, in general, individuals can only recall 7±2 (between five and nine) items (Mandel, 1997). In addition, the storage time of information within short-term memory is limited to 30 seconds (Galitz, 2007; Mandel, 1997). When short term memory is full and once this period of time has elapsed, if any new information arrives, the short term memory will retain the latest information instead of the previous information. Some authors refer to short-term memory as working memory. However, the term working memory is reserved for the processes and structures in the brain that manipulate or modify memory. Nevertheless both terms are often used interchangeably. Techniques exist to overcome these deficiencies in short-term memory. Two such techniques used to enhance the performance of short term memory and extend the period of time information retained are (i) Rehearsal and (ii) Chunking (Human-memory, 2010b).

- Rehearsal (say it aloud). A constant repetition of the fact to be remembered will result in its repeated arrival into short term memory via the sensory system. This constant "refreshing" will lead to the information being retained for a longer time in short-term memory.
- Chunking (grouping items), this is where to exploit patterns in order to extend recall of information. For instance when you hear or read a phone number and you want to remember it for longer you can group it like this 07779-414-282.

The third type of memory is referred to Long-Term Memory. Long-term memory has unlimited capacity for information storage and on its name suggests this memory keeps the information for a long-time. Long-term memory contains all the active "processing"

knowledge collected from short-term memory and as such it has a large amount of storage available. Long-term memory is used to store the activities of an entire human life. Long-term memory made up of three parts (Human-memory, 2010a):

- The first is called semantic memory. Semantic memory holds facts, knowledge and information about the external life.
- The second is procedural memory. This type of memory holds the skills of how to do things. For instance how to ride a bike or how to drive. Procedural memory is all about practically dealing with objects and body movement.
- The final type of long-term memory is called Episodic / Autobiographical memory. This type of memory holds all the details of personal events, as well as a person's history and experiences. For instance the date of student graduated or the date of man/ women gets married.

Back in 1974, Ted Nelson has defined "the ten minute rule". This rule stated that "Any system which cannot be well taught to a layman in ten minutes, by a tutor in the presence of a responding setup, is too complicated" (Usability First, 2015). This means that any novice user should not take more than ten minutes to learn a new system. However this could not work with some of complex systems of today. For instance, pilots who use airplane simulations system could not expect to learn how to operate such a system in so little time. Nevertheless, Nelson's rule was successful in raising the issue of "learnability" (Ccit333, 2015; Preece, Rogers, & Sharp, 2002).

2.2 Interaction

Users are a part of most systems and the experiences they have when interacting with the system will have either a positive or negative impact on the performance of the system as a whole. Juristo et al.(2007) stated that interaction is "the coordination of information exchange between the user and the system" (Juristo et al., 2007). This section of interactions describes the four main topics which make up the topic. These are: interaction components, activities of interaction design, interaction types, and interaction style.

2.2.1 Interaction components

Every system is designed using a number of components which interact with each other. These components are users, the tasks they seek to complete, the context / environment and other system constraints. Thus, the designer needs to be careful about each of these

components in order to evolve a usable design. Designers need to know who the system users are and they need to understand some of their characteristics. Designers need to identify what the users want to achieve (user goals), why the system is being developed as well as what type of tasks will performed by the system. The designer should determine details of the environment and the context within which the systems will operate. They also need to define the technical and logistical constraints for using the system. Details of interaction between these components are required in order to evolve a system that will be usable. when these components are considered the system is more likely to support intuitive interaction (Preece, Benyon, & University, 1993)

2.2.2 Activities of interaction design

The system designer needs to carefully craft the user's interaction with the system. Preece et al. (2002) identified the following activites that should be performed when designing the user's interaction with the system:

- Identification of user needs and requirements
- Developing alternative designs
- Building the design and
- Evaluating the design

Note, these tasks should be completed during the design of the user interaction, not during system implementation. Each of these tasks will be examined in turn.

Identification of user Needs and requirements: requirements analysis and specifications is a huge area of software engineering with a vast quantity of resources and techniques. Fundamentally, users are the target of the product design. Therefore, there should be a clear understanding of their needs, and what support they require from the system. A specific list of requirements should be defined at the beginning of the product design. A number of methods can be used to collect the user needs and requirements. For instance, use of focus groups, user's interviews and questionnaires.

Developing alternative designs: means the preparation of alternative solutions that meet the requirements. Developing alternative designs is important in order to help the user steer the design. When the design comes up with a number of alternative solutions, users can select their preference rather than having articulated them. There are two methodologies which can be used to generate alternative designs. The first is known as the "flair and creative methodology" which means the designer creates new alternative designs after researching the

topic and coming up with entirely new solution. The second methodology known as the "inspiration methodology" which means the designer looks at other designs and seeks inspiration from them(Preece et al., 2002).

Building the design: At this final stage of the design process, interaction design should be implemented. The means by which the user will interact with the design system should be design and clear specified at this stage, note that this does not necessarily require software development. However it can involve using any techniques that will achieve the interaction design. For instance, it is possible to use a paper-based prototype. This part of the process is often an iterative process as the designers seldom get the design correct the first time. The prototype developed at this stage should be sufficient to accurately portray the user interaction. A formal software development methodology can assist on this stage process. For instance, Spiral software development model.

Evaluation design: At this stage of the design process, usability and accessibility issues will be determined. User errors and how to redesign and solve them will be determined by combining requirements. Both end user and experts can be employed to evaluate the design. There are many evaluation techniques can be employed to evaluate the design. For instance User testing and heuristics (Preece et al., 2002).

At the end of these four objectives, the user interaction should be clearly designed.

2.2.3 Interaction types

User interaction generates "inputs" to the computer system. These inputs are then processed and the results are the "output" which is presented to the user. User interaction types were described by(Yang & Chen, 2009) and they mentioned that user interaction is created based on four types of data. These are:

Data Interaction is the input and exchange of the data. Figure 2-1 describes the process of data interaction.

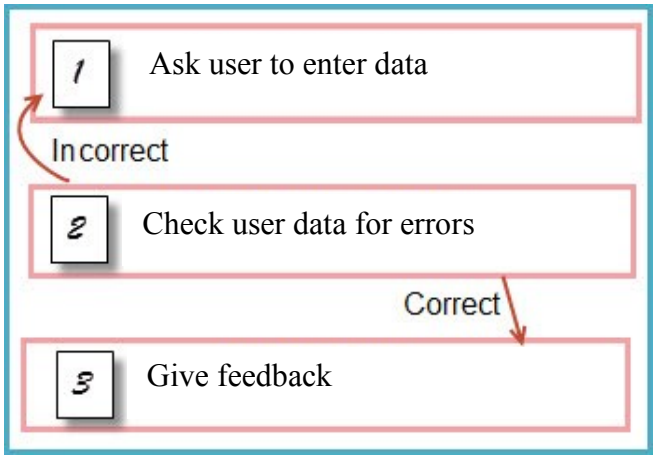


Figure 2-1 Data Interaction Process

Image Interaction is "the computer's understanding of an image based on human behaviour." The human sensory system is based on a number of different processes; however images may account for more than 70% of received information from the system (see Figure 2-2).

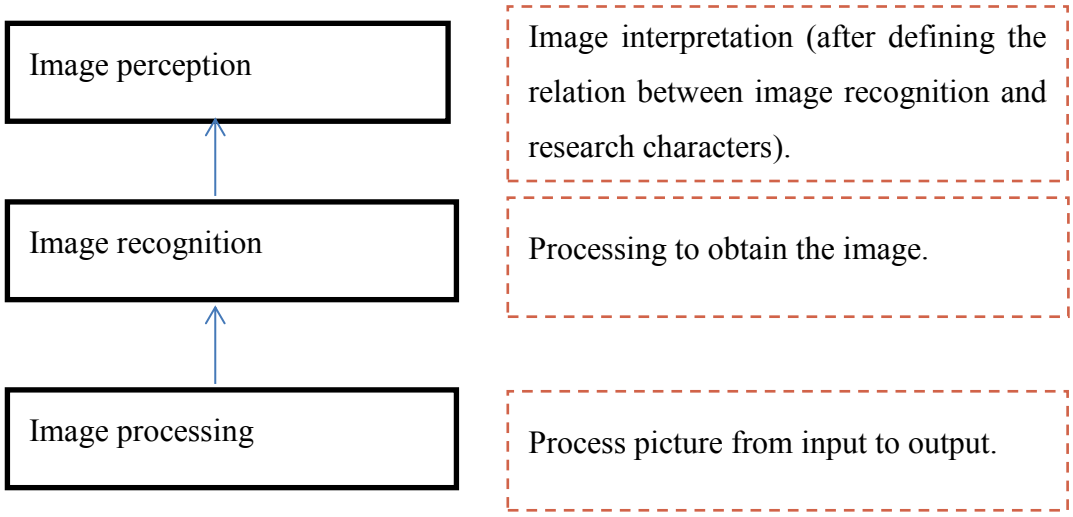


Figure 2-2 Image Interaction Machine Vision

Aretinal scan is one technique used to relate machine vision to interaction. This technique exploits human biometrics, by a set of metrics which are unique to each individual to identify and validate personal access to restricted areas. For example, the Retinal scanner is used in high-security locations.(Wikipedia, 2015)

Voice interaction is the use of natural language (voice) to interact with the computer. It is considered to be an important mode of interaction, since an auditory signal is faster than a visual signal dictation. Voice interaction style system can exclusively use pure voice controlled interactions for inputs. Alternatively, hybrid system, which use voice input controlled with other interaction type are also available.

Behavioural interaction is a form of human action interaction. In behavioural interaction, the system interprets human movements by following a system and organising the human behaviour within the system. The output is intelligent feedback. For example, this type of interaction is applied to an automatic door that open as a person approaches it. This output (door opens) is the result of interpreting the human behaviour (approaches door).

2.2.4 Interaction style

The interaction style is the manner of communication between user and computer (system). Moreover, the interaction styles are used to create different types of interface considered to be the interaction environment for users; however, HCI is more concerned with how these interfaces create interactions. Thus, designers should be more focused on interaction and not only interface design (Beaudouin-Lafon, 2004). Nowadays, the graphical user interface (GUI), speech recognition interface, the tangible user interface (TUI), organic user interface and eye movement interface are the future of interaction interfaces as these have created a new interaction style that allows users to interact with interfaces that simulate interaction with the real world (Ishii, 2008; Jain, Lund & Wixon, 2011; Vertegaal & Poupyrev, 2008)

Interaction has many styles. However, five common styles have been identified by Shneiderman (1991) (Dix et al., 2004; Shneiderman, 1991; Stone et al., 2005) These types of interaction style include command line style, menu style, forms style, direct manipulation style and anthropomorphic style. A description and example of each style is given for clarification.

One of the most common interactive interfaces dealing with data is the command line interface. This interface requires the user to enter a command and then the system presents the result. For example, when the command "dir" is typed at the MSDOS command prompt, the user send the command to the system by pressing the enter key. The system responds by displaying a list of all directories and files in the current

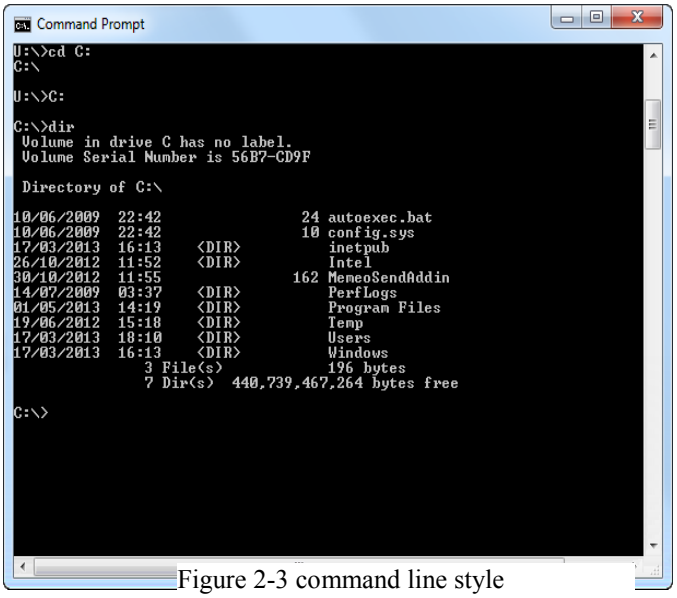


Figure 2-3 command line style

directory.(See Figure 2-3)

Menus are another common style of interaction. Menu used to avoid the errors made using a command line interface. This style of interaction allows the user to interact with numerous items from a menu. Menus are required to be visible to the user. However, menus might begin in an invisible state, for instance when a user right-clicks an icon to access to popup menu.(See Figure 2-4)

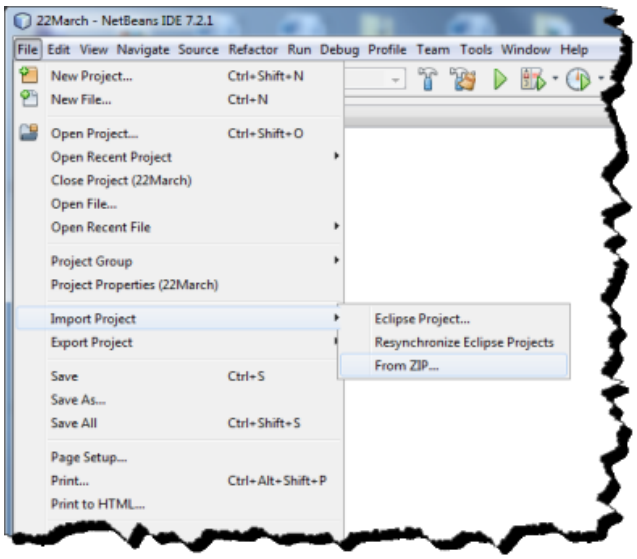


Figure 2-4 Menu style

Forms are another interactive style used to collect information from the user. The user should complete the form in order to interact with it. (See Figure 2-5)

Figure 2-5 has been removed due to Copyright restrictions.

Direct manipulation is another interaction type enables the user to interact directly with the interface objects. Direct manipulation is applied in many applications. For instance, direct

manipulation is used in file organisations tool such as windows explorer. In explores a mouse is used to drag a folder from one place to another; thus, direct manipulation occurs. For direct manipulation to be effective, devices must present a "smooth or continuous" response. For instance: a mouse device must create a smooth and constant motion of the mouse pointer.

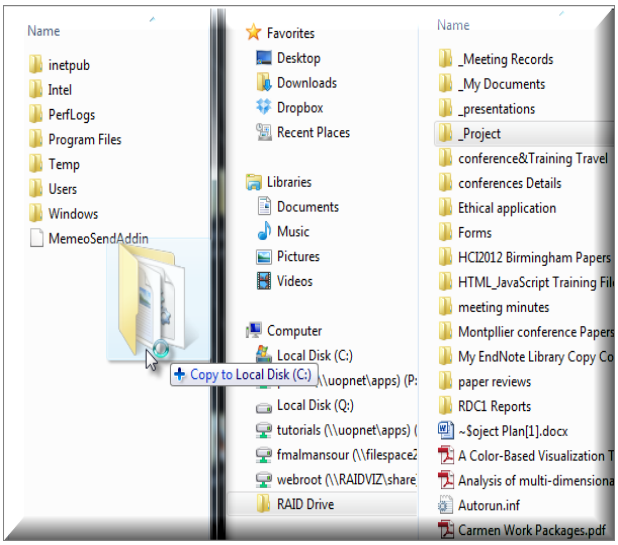


Figure 2-6 Direct manipulation style

Direct manipulation is not limited to 2D data representations; it has recently been extended to the 3D world. There are known as Tangible User Interfaces (TUI) where the user directly manipulates 3D objects to control the digital data and processing inside the computer. An example of this is shown in Figure 2-7 where an electronic musical instrument is played (programmed) using 3D objects.

Figure 2-7: has been removed due to Copyright restrictions.

Anthropomorphic also is another interaction style. The main purpose of this style of interaction is to make input more natural. Fully understanding human behaviour and how people contact and interact with each other in real time is vital in order to effectively implement an anthropomorphic style interface. Eye movement interfaces and natural language interfaces are examples of the anthropomorphic style of interaction.

2.3 Summary:

HCI is considered an important subject of which all software engineers should be well informed in regards the important elements. Thus, after a review of the significant literature in relation to the aims and objectives of the study, it can be said that three key elements should be taken into account. These are human (user), design (product) and the interaction between the user and design. The researcher's clear understanding of human properties is an essential way of increasing the level of productivities and the usefulness of the product, as in the case of human memories and cognitive capabilities. Furthermore, the researcher also is required to adopt an interaction approach between users and product. Thus, given that these five interaction styles exist, it is natural to query whether it is possible to combine different interaction styles in one interface design. Importantly, the answer is yes. There are many user interfaces that combine different interaction styles, especially for interfaces, with a wide range of uses: for instance, the menu style can be integrated with a direct manipulation style to support a high level of user control (Hix & Hartson, 1993). However, the integration of different types of interaction style should be carefully employed so as to avoid distracting users (Stone *et al*., 2005).

In this study, researcher was concerned with human factors, and therefore reviewed the existing practice, including research results, for human proprieties of concern, such as reducing the short-term memory load by reducing the length of information at each interface, as well as involving users during the product-creation process so as to create an appropriate level of interaction between user and product.

Chapter Three: Software Usability

This chapter provides the concept of software usability and how to design usable products that are easy to use, before presenting a view of measurement elements and methods employed to assess the usability of the product. Interpreting the assessment results, it is important to understand the level of usability. Finally, sum-up the chapter with ‘Discount usability’ term and how it’s impact the software usability.

3 Software Usability

Numerous definitions of usability exist, with the extent and depth of the topic much debated: for instance, usability is the capability of something to be used by humans easily and effectively where 'easy' equates to a specified level of subjective assessment and effective means of a specified level of human performance (Shackel, 2009b). Usability may be defined as 'the degree to which people (users) can perform a set of required tasks' (Brinck, Gergle & Wood, 2002). Originally, the ISO 9241-11 referred to usability as '*the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specific context of use*' (Ramli & Jaafar, 2008). These differing definitions largely reflect the author's overall vision of usability and their working fields. However, the ISO definition is the most commonly applied owing to the fact it covers a wide range of attributes, including the effectiveness, efficiency and level of satisfaction associated with the product.

As far back as (Preece et al., 2002), several usability goals have been recognised, noting that, if one sought to build usable and associated software, these objectives needed to be carefully considered.

The goals are as follows:

1. Effectiveness: The software should do what is expected.
2. Efficiency: The software should enable users to learn and carry out their tasks. It should also determine how long it will take to complete the tasks.
3. Safety: The software should protect users from harm or data loss.
4. Utility: The software should deliver software tools and facilities that help users to do what they want to do.
5. Learnability: The software should be easy to learn and use. According to Nelson (1980), the novice user needs only ten minutes to learn about software or the system ultimately will fail. However, with a complex system—such as a pilot flight system method, for example—this may be impossible (Preece *et al*., 2002); therefore, learnability should be measured with respect to complexity but, most importantly, should not be too onerous on the user.
6. Memorability: The software should be easy to remember so the user can recall how to undertake the same task again.

Overall, according to (Preece *et al*., 2002), the software must be able to achieve all of these goals in order to be referred to as 'usable'.

Overall, each software product is built to match a list of benefits specified by the software engineer and the commissioning organisation; however, 'the benefits of better usability are not easily identified or calculated' (Rajanen, 2003). This difficulty could be owing to the usability being a wide area and employed by different researchers from different fields; thus, each researcher has a list of benefits. For example, Maguire(2001) lists a number of benefits underpinning what he may deem a usable design (M. Maguire, 2001). The combination between these different benefits will affect the usability on different aspects, including the productivity of software, reductions in costs of training and engineering, and users' satisfaction levels.

Incredibly, usability has been incorrectly classified as a part of software development attached or added on towards the end of the development cycle; this is often misunderstood as being part of the 'finishing' of the product, which is totally incorrect(Dicks, 2002). Usability is core to the success of the software. Usability is central to efficiency and throughput, and thus it is core to business and software development(Nielsen, 2012).

3.1 Usable Design Principles

Usability research shows that usable high-quality functional software requires outstanding design, as well as the observation of numerous design principles. Many authors have published research in this area, with many recommending the use of principles such as Mandel's Three Golden Rules (Mandel, 1997). Note that each rule also contains its own principles. These are discussed as follows:

(i) Place user in control: Allow the user be the system driver, controlling how and where they interact with the system. The system should be able to manage many levels of user, enabling them to seamlessly interact with the system in a manner based on their control and experience. Note that Mandel also lists several principles under this rule centred on achieving the goal of this rule. Furthermore, all of these principles can be described in ten short words: modeless, flexible, interruptible, helpful, forgiving, navigable, accessible, facilitative, preferences and interactive.

(ii) Reduce user's memory load: The rule is focused on the cognitive memory loading of the user as they use the system. All design should consider the limitations of memory load for each human being. All principles under this rule are described with the

following short words: remember, recognition, inform, forgiven, frequency, intuitive, transfer, context, and organise.

(iii) Make the interface consistent: Mandel identifies consistency as a key aspect of usability. Consistency in software enables users to exploit their knowledge and facilitates learning between programs. The principles of interface consistency can be described in the following few shorts words: continuity, expectation, attitude and predictability.

Nielsen (Nielsen, 1993a) coined his own key aspects of usability when he listed the following attributes to identify the usability of a system:

(i) Learnability: Software design should deliver a system that is easy to learn.

(ii) Efficiency: Software design must produce a system that will enable the user to be efficient, thus ensuring a high level of productivity.

(iii) Memorability: The design must be easy to remember, both between interfaces as well as after a period of time without using the system.

(iv) Low error rates: The system should expect some errors during the user's works because this is a way of recovering from errors; however, such errors should not be significant.

(v) Satisfaction: Software should be accepted by the user and should satisfy the user.

Many researchers share similar views, although these may be expressed in different terms and taxonomies. For example, (Stone et al., 2005) present their taxonomy and various principles, such as simplicity, structure, consistency and tolerance. Although, Shneiderman lists a number of design principles referred to as 'Eight Golden Rules', where these 8 principles are considered as guideline for software design (Shneiderman, Plaisant, Cohen, & Jacobs, 2013). These are as follows:

1. Strive for consistency
2. Cater to universal usability
3. Offer information feedback
4. Design dialogs to yield colure
5. Prevent errors
6. Permit easy reversal of actions
7. Support internal locus of control and
8. Reduce short-term memory load.

3.1.1 Achieving Good Design (Distilled):

Norman (2013) mentioned that discoverability and understanding are the most important two elements for creating a good design. Discoverability means the ability to know what the possible actions that design could produce. Understanding means an overall understanding of the design works and use(Norman, 2013)

In order to distil these principles into a manageable list of helpful principles for the developers, the principles of two core researchers (Gong, 2009; Shneiderman *et al*., 2013) were synthesised alongside other principles. The outcome was the following list of 10 guidelines for good design:

1. Good design considers human factors. This means that the end product will facilitate interaction, communication and understanding as the software designer will integrate the knowledge of these factors. In addition, these factors should be universal for all users.
2. Good design incorporates the stability of both design and information. Design should cover user support and user guidelines, and should propose problems to solutions in order to avoid mistakes being made by the user.
3. Good design should integrate consistency. Using the same words and situation for each interface of the design will deliver strong consistency across the software. However, different words or situations with the same reaction will be difficult to understand, learn, or be dealt with by the user.
4. Good design should have no omissions whilst also being concise. Designers should avoid including unnecessary information, and should base their system on an intuitive approach. Concise development will improve user learning and an understanding of the aims of the application.
5. Good design should have flexibility. Software is controlled by the user, meaning flexibility affects how the user works and their overall capacity for control. Flexibility means providing multiple options, which will enable the user to choose their preferred approach and thereby exercise control over the end product.
6. Good designing, in part, relies on memorability. Using familiar words, images and icons improves users' understanding of the interface. Classification, taxonomy and categories with the same objects or function are required within the design, meaning it then will be easy to memorise, thus reducing the possibility of errors.

7. A good design will have predictability. Experience can guide the user to predict the results in advance if predictability is effectively incorporated within the design.
8. Good design should also meet users' emotional needs, i.e. intuitive thinking, humour and appropriate challenges.
9. Good design also allows for assistance and support. User guidelines are an intrinsic element of the package, helping to improve users' ability to understand the software and accordingly recognise relevant guidelines.
10. Good design should provide for users' needs, with multimedia technology used to meet these needs.

Finally, these principles considered as the basic elements for creating a software design, thus assessing these principles should be taken on the next phase for the software development process. Usability evaluation is essential topic that software engineers required to understand and apply for evaluation the design.

3.2 Usability Evaluation

The American Evaluation Association (AEA) defines evaluation as 'a systematic process to determine merit, worth, value or significance' (Administrator, 2014). This definition remains based on the original, and is defined as the process of assessing the user experience of a software system. In essence, design evaluation is focused on discovering any issues within the user experience, so as to allow the software to be improved in order to increase user productivity (Karat, 1997). Evaluation is the means by which evaluators assess: (i) the quality of the software, (ii) the usability of the system, (iii) the extent to which the user's requirements have been met, and (iv) the ability to identify system problems. Note that the latter is based on user satisfaction with the system (Dix et al., 2004; Stone et al., 2005).

This stage of the software development lifecycle is crucial for many reasons. Firstly, the communication of requirements between users and developers is a difficult task, despite the numerous methodologies that exist to support this process. Thus, it is essential that checks are carried out so as to ensure the product matches user expectations. It is equally important to ensure that the developer has an in-depth appreciation of how the user intends to use the software, as well as any limitations of the software. Thus, it is a very worthwhile and important stage of the software development lifecycle (Preece *et al*., 2002). Number of authors have stated that usability evaluation is significantly impacted for many factors; these are (Carmelo Ardito et al., 2011; Bias & Mayhew, 2005):

- Quality improvement of the developed product
- User satisfaction with the product
- Increase the organisation competiveness
- Resource saving (reduce the overall cost)
- Increase the product sales and user's productivities
- Reduce the training and user support cost

Furthermore, it is impossible for designers to judge their own software even when such judgements are made in conjunction. It is much more satisfactory for such objective assessments to be made using a list of criteria. However, it is not enough solely to solely follow design standards and guidelines; rather, it is essential to assess the software under development and preferably to do so numerous times during development—not only at the end of the design (Holzinger, 2005). Therefore, it is very well established that it is much easier and less costly to fix errors early on in the development process. Finally, testing and evaluation are essential as they make up the final stage in establishing that the software is complete and fit for purpose (Galitz, 2007)

3.2.1 Measurement Elements

Van Velsen *et al*. (2008) found that usability was the most measured variable (van Velsen, van der Geest, Klaassen, & Steehouder, 2008). Literature provides many of design measurement classifications that established by different authors. Table 3-1 shows three chosen classification from the literature. Nielsen (1993) assumed that products should be measured by running a number of tests in order to assess (i) the ease of learning about the product, (ii) the efficiency of use, (iii) the ease of recall, (iv) the number of errors, and (v) the ease of use (Nielsen, 1993a).

Folmer & Bosch (2004) lists thatISO9126-1 mentions several measurements that will provide the meaning of usability. The measurable attributes are as follows:

- Understandability: The extent to which software is understandable for users.
- Learnability: Refers to the ease with which users can learn to use the software efficiently.
- Operability: Refers to the extent the software is operated and controlled by the user.
- Attractiveness: The extent to which the software is attractive to the user, such as in terms of the use of colour in its design (Folmer & Bosch, 2004).

Shackel (2009) mentions four key areas to usability testing: these are learnability, throughput, flexibility and attitude. Learnability is a metric based on the ease of the system in terms of its ability to be learnt and understood. Throughput refers to specific metrics of task completion, including the speed of execution, and the possibility of error and mistakes. Flexibility refers to the ability of the system to support different levels of user interaction based on users' experience. Finally, attitude refers to the system's overall lability to give the user a positive experience (Shackel, 2009b).

Table 3-1: Review of design measurement elements

Author	Measurement Elements
Nielsen (1993)	- Learnability - Efficiency - Memorability - Errors - Satisfaction
Folmer & Bosch (2004)	- Understandability - Learnability - Operability - Attractiveness
Shackel (2009)	- Learnability - Throughput - Flexibility - Attitude

Finally, there are many different taxonomies that have been established so as to explain the usability measurements, where each taxonomy is correct and able to be considered as measurement elements. The goal of this study is to minimise the wide argument centred on the usability definition and measurement elements. Thus, the ISO usability definition has defined three main keywords (effectiveness, efficiency and satisfaction) that can be a comprehensive explanation for the usability measurement(Jokela, Iivari, Matero, & Karukka, 2003). Additionally, It has already been established (Beven, 2006) that effectiveness, efficiency and satisfaction form ISO usability definition are key to the practicality of testing measures design. Effectiveness ensures the design achieves its goals; that it is free from

errors with heightened accuracy. Efficiency means the time and effort to achieve these goals is less, and also implies speed. Satisfaction relates to users' impressions and feelings regarding the usefulness of the design; in other words, whether or not they are happy and whether they accept the product in its current format (Bevan, 2006; Patrick W, 1998). Thus, evaluators should concern about the ISO usability definition to establish their own usability measurements which are required to measure by using usability evaluation methods.

Furthermore, Beavan (2008) mentions that 'UX can be measured as the user's satisfaction with achieving pragmatic and hedonic goals, and pleasure. (Bevan, 2008)'; this is related to ISO definition.

3.2.2 Usability Evaluation Methods (UEMs)

Evaluation methods are required to identify the usability problems of software products (Henderson, Podd, Smith, & Varela-Alvarez, 1995). One of the most highly respected specialists in the area of design evaluation is Nielsen (Nielsen & Mack, 1994), who asserts that empirical methods (those that directly deal with participants) are commonly used, even though it can be difficult to recruit participants. User testing is effective in identifying problems that are difficult to establish using other methods; however, Nielsen is pragmatic, and therefore accepts that, in situations where it is too difficult or too expensive to recruit participants, inspection methods can be used. Inspection methods incorporate a range of specific guidelines or principles that may be used to reveal problems. In 2006 Ardito et al. found on their study that evaluators rated the user-centred methods higher than inspection methods with favour to satisfaction of use (Carmelo Ardito, Costabile, De Angeli, & Lanzilotti, 2006). Accordingly, in practice, a combination of empirical and inspection methods can achieve accurate evaluation results (Nielsen & Mack, 1994; Shneiderman *et al*., 2013). Furthermore, Van *et al.* (2008) have found that most of their studies reviewed in a personalised system used more than one method during the evaluation process (van Velsen et al., 2008). Moreover, In 2008 Frøkjær and Hornbæk found that evaluators are more prefer using users involving methodology over the inspection methods(Frøkjær & Hornbæk, 2008) . The literature shows that many authors established different classification of usability evaluation. In 1994, Nielsen defined four main classes of evaluating user interfaces:

Automatic evaluation: The usability of an interface is measured by submitting the interface as input to a software application.

Empirical evaluation: The usability of an interface is assessed by testing the interface with participants.

Formal evaluation: The interface is assessed using exact models and formulae to calculate usability measures.

Informal evaluation: An assessment is made by an evaluator using rules of thumb and the general skills and experience of the evaluator (Nielsen & Mack, 1994).

Some years later, Jacobsen (Jacobsen, 1999) challenged this by proposing an alternative classification of evaluation methods. Jacobsen divided the evaluation into three main categories. Firstly, *empirical evaluation* which is user-centred evaluation process is adopted throughout the development of the product. Secondly, *inspection evaluation* which is an expert inspects and reviews the system using established guidelines and principles. Thirdly, *inquiry evaluation* which is user feedback is collected and reviewed before, during and after the evaluation process.

Dix *et al*. (2004) classified the evaluation methods into two groups where each group is based on the experience and expertise of the participant. These classifications are as follows:

1- *Evaluation through expert analysis*: Expert evaluators assess the product design using a combination of different methods, namely (i) cognitive walkthroughs, (ii) Heuristics (a list of principles underpinning good design practice, as defined by Nielson (1995a), (iii) other specific models, such as the GOMS (Goals, Operators, Methods and Selection) model, and (iv) using previous work results.
2- *Evaluation through user participation:* Users assess the product using methods such as (i) empirical experiments, (ii) observations, (iii) querying using questionnaires, and (iv) physiological responses, such as eye-tracking (Dix *et al*., 2004).

This review of evaluation methods identifies three robust classifications of design evaluation, each of which is defined by eminent researchers in the field. Across these three classifications, debate continues on the advantages and disadvantages of evaluation. Numerous additional classifications of design evaluation exist; new classifications and methodologies continue to evolve. The variety of evaluation methods available to the software developer is large and can be complex to navigate. Thus, it's essential for developers to understand the goals of performing the evaluation prior to choosing the evaluation methods (Dix *et al*., 2004). However, it can be difficult for developers to establish which evaluation methods are the most efficient and useful for evaluating their

software(Carmelo Ardito *et al*., 2011). Developer preferences may be viewed as one of the predominant obstacles hindering the development of usable software, as in the case of the mind-set of the developer. It has been stated by (C Ardito et al., 2014) that there are three key obstacles effecting developers' ability to create usable software and carrying out usability assessments, including development mind-set, the wealth of resources necessary to complete a usability evaluation, and the problems and complexities involving users in the usability evaluation process. Accordingly, various efforts have been made by developers to avoid users' participation in the development stages owing to the view that such involvement can waste time, may mean unrealistic requests, and the uncertainty of users concerning their needs (C Ardito et al., 2014). Moreover, the lack of usability evaluation knowledge is one of the key issues facing developers in the completion of the usability assessments on products (Rosenbaum et al., 2000). Accordingly, the creation of software with a lower level of usability evaluation knowledge and without the direct participation of users in the process of development can mean developers create products based on their viewpoints and own experiences. As such, some products following the completion of development.

However, developer training proposed as solution to increase the developer awareness of user involvement and evaluation conduction. Thus, Number of previous studies identified that inexperienced usability evaluators are able to conduct the usability evaluation by using tools, training or learning resource to come up with list of identifying problems (Bruun & Stage, 2014; Howarth et al., 2009; Skov & Stage, 2005). Furthermore, in 2012 Skov and Stage conducted a study to investigate the student ability of conducting evaluation after they have training course. This study provided 234 of first-year undergraduate students with 40 hours of training. as results of this experiment students "gained good competence in conducting the evaluation, defining user tasks and producing a usability report, while they were less successful in acquiring skills for identifying and describing usability problems" (Skov & Stage, 2012). The previous review shows that evaluation training could be a great solution to improve the non-expert evaluators confidently of conducting the usability evaluation.

In this research, overall dEv tool outcomes support that programmers can be encouraged to avoid the mindset perspective. Thinking as a user and developer is one meaning behind mindset (Bak, Nguyen, Risgaard & Stage, 2008). Thus, in Chapter 9 (Clock Study), it is clearly shown that dEv groups were beginning to avoid thinking as both a user and developer; thus, they involved more users and evaluation methods than the other two groups. This means that the dEv tool participants are willing to involve users on the development process phases

by using different evaluation methods and accepting different feedback and views relating to the design. Thus, this leads them to consider themselves as developers more so than users.

There are many common evaluation methods as follows:

Heuristics Evaluation

This method is used to evaluate the interface design against list of principles. Heuristic evaluation was established by Nielsen & Molich in 1990, with Nielsen in 1994 setting the of heuristics with more explanation (Nielsen, 1995). This method provides a set of 10 heuristics that discover usability problems. These are:

1- Visibility of system status
2- Match between system and the real world
3- User control and freedom
4- Consistency and standards
5- Error prevention
6- Recognition rather than recall
7- Flexibility and efficiency of use
8- Aesthetic and minimalist design
9- Help users recognize, diagnose, and recover from errors and
10- Help and documentation

The method conducted by expert evaluators rather than end users also is recommended for completion by multiple evaluators rather than one evaluator, with Nielsen stating that 57% of usability problems can identified with the use of five evaluators (Dix *et al*., 2004; Nielsen & Mack, 1994). Heuristic evaluation is easy, inexpensive and quick, and does not require advanced planning and application in the early stages of the development process to identify a lot of usability problem (Nielsen & Molich, 1990). However, the expert evaluators with HCI experience are recommend to perform the evaluation method (Galitz, 2007). Furthermore, evaluators with less or no experience are able to perform the HE; however, results are not as good expert evaluators (Holzinger, 2005).

Cognitive Walkthrough

This method is used to *walk through an interface in the context of representative user tasks (Galitz, 2007).* The cognitive walkthrough is considered an alternative method of the heuristics evaluation method and running between the developers in the development team

without any user involvement (Preece *et al*., 2002). The main concept of using cognitive walkthrough is centred on creating a list of actions that are required to complete a specific task with real users. Dix *et al*. (2004) mentions four things that are required for running the cognitive walkthrough on the system, namely:

1- System prototype with clear detail for interface components
2- Task descriptions that users perform in the system
3- List of written actions to be followed to achieve task completion
4- User preferences, including level of experience (Dix *et al*., 2004).

The cognitive walkthrough provides a clear evaluation process, inexpensive method, no need of system functions, and can be run by the developers themselves. However, it is a boring method for evaluators, and there is the possibility of bias in the task-selection, with its developers not requiring user involvement (Galitz, 2007; Holzinger, 2005).

Usability Testing

This method is used to test the interface or product in the real-word by involving real participants. Normally, participants will ask to complete a number of tasks during the usability session. Testing can be done for the design function testing or for design requirements, checking whether or not functionality is met (Shneiderman *et al*., 2013). The results of the usability testing are used to improve design functions or performance (Riihiaho, Nieminen, Westman, Addams-Moring & Katainen, 2015). Owing to the fact that usability testing is performed in the real world, this method is able to provide a number of benefits for the design developer (Preece et al., 2002; Usability.gov, 2013c). These are:

- Recoding the time of task completion for each participant and accordingly improving it,
- Recoding the number and types of error detected by participants,
- Measuring each task, and
- Identifying the problems preventing participants from completing the tasks.

However, usability testing requires expensive requirements, including an expert investigator with user interface experience. Furthermore it is not appropriate for identifying inconsistency problems (Galitz, 2007). Inconsistence problems are produced in the designing phase as a result of missing the application standard for the interface design, such as in using different words for the same things (Nielsen, 2008). Thus, usability testing it is not appropriate for

identifying inconsistency problems as dealing with real users who have different levels of experience will focus on low-priority items and find issues related to specific tasks (Galitz, 2007; Halabi, 2007).

Thinking Aloud

This method is used to encourage participants to speak aloud during the task performance. The thinking aloud method helps investigators to establish what participants are thinking about when they doing the tasks, and also allows the opportunity to discuss various points with them (Patrick, 1998). Normally, thinking aloud is individually performed with single users; however, the co-discovery method is a way of involving two participants using the system at the same time. The co-discovery method is aimed at letting participants teach each other and solve the encounter problems (Holzinger, 2005). Moreover, the think aloud method is important as it is a way of keeping developers from asking themselves why users do this and allows them to establish answers from the participants during the task performance. Additionally, it is ways of ensuring participants are more constraint with the system. However, there are some challenges of using thinking aloud method, for instance some participants cannot talk during the evaluation sessions as they are shy or it is difficult hard to talk and perform the task at the same time. Furthermore, the thinking aloud technique may slow down the process of participants' thinking. Therefore, this method prevents the collection of data under the circumstances outlined (Galitz, 2007; Holzinger, 2005; Patrick, 1998). Finally, thinking aloud can be used to get useful feedback about users' behaviour and usability problems (van Velsen et al., 2008)

Focus Groups

This method is used to discuss the design process, prototype or requirements of a group of users or developers. Focus group sessions are normally between 3 and 10 participants who sit together and discuss with the design team (Preece *et al*., 2002). It is a valuable method for encouraging participants to talk and provide their opinions and feedback about the session subject. Thus, the focus group method provides a large volume of data; however, expert moderators are required to run the session correctly and effectively (Galitz, 2007).

Scenarios

This method is task description or steps that ask users to follow through the design. Scenarios can be used for both the design and evaluation of the interface. The scenario method is a way of identifying who the participants are that are working with the software design. Scenarios can be used during the usability testing by involving tasks; however, these tasks should not detail who completes these tasks (Dumas & Redish, 1999; Usability.gov).

Observation

This method is used to observe the participants in the use of the interface. It is considered a valuable method between all qualitative data collection methods (C. Marshall & Gretchen B, 2006). Observations can be implemented by vising the participants at their real work location and watching them without asking them to perform any particular tasks (Holzinger, 2005), or otherwise by observing during a set of tasks (Dix *et al*., 2004). The observation method is important when seeking to determine users' actions and reactions to the interfaces. Recording the data can be achieved through different techniques, including paper and pen, video, audio, computer-logging and notebooks (Dix *et al*., 2004).

Timing and Logging

This method is used to time the user activities on the system performance. Timing and logging methods help the developer to measure the task accomplishing time. The timer could be recoded either manually (stopwatch) or automatically (automated tool). Automated tools are important in measuring the complex software by tracking participants' actuations, time, mouse clicks, login and so on. Automated logging tools are expensive; thus, clear understanding of the project evaluation objectives is required before using one of these tools (Patrick W, 1998; Stone *et al*., 2005).

Questionnaire

This method uses a formalised set of questions to gather information from participants. The questionnaire method is one of the most common evaluation approaches used to measure usability, perspective usefulness and intention to use (van Velsen et al., 2008). There are two techniques used to complete the questionnaire: firstly, giving the participants freedom of time to complete by themselves and return the questionnaire; and secondly, interviewing the participants either face-to-face or by telephone, and asking them to complete the questionnaire. The questionnaire method is a way of collecting both quantitative and

qualitative data, and is inexpensive and quick to gather form large numbers of participant. However, the questions are difficult to design and need to be tested on a small group of participants before sending out to a large number of participants (Patrick W, 1998; Shneiderman *et al*., 2013; Stone *et al*., 2005).

Interview

This method is used to ask participants a number of questions and garner their answers in an individual meeting. The interviews are a way of discovering the research subject through the use of deep information. Thus, the investigator should prepare and plan for the interview, and the questions should start as more general and then become more specific(Dix *et al*., 2004). There are three types of interview structure: firstly, unstructured interview, which is appropriate if the investigator does not have enough of a clear idea about the subject and plans on asking a number of questions; secondly, semi-structured interview, where this type is used if the investigator has a clear idea of the interview subjects and wants to cover these questions, where the number of questions may increase during the interview; and finally, structured interview, where all questions are pre-set and each interview is asked exactly the same questions in the same order (Patrick W, 1998). The interview method is a great technique for gathering a large volume of qualitative data; however, it is expensive and time-consuming (Shneiderman *et al*., 2013).

3.2.3 Discount Usability

Discount usability has been proposed as a term by Nielson (Nielsen, 1993). Discount usability aims at employing high-speed and low-cost evaluation methods in mind of assessing and improving software usability. Moreover, discount usability proves that inexpensive methods—not only expensive methods—can impact the level of software usability (Nielsen, 1995b). There are three methods that are suggested as discount methods, namely scenarios, thinking aloud and heuristics evaluation (HE) (Nielsen, 2009). Thus, the discount methods are useful methods for getting quick results of the software design, especially for developers who do not have that much experience in software evaluation. Furthermore, school students are recommended to learn about usability via discount usability as it is a valuable initial concept (Nielsen, 1997).

3.2.4 Automated Evaluation Tools

Nowadays, automated evaluations have become a common tool for assessing software. Such a method is easy to use and gets results, which makes these tools commonly deployed. Automated evaluation has been defined by Ivory & Chevalier (2002) as 'software that automates the collection of interface usage data (automated capture) or the identification (automated analysis) and the resolution (automated critique) of potential problems' (Ivory & Chevalier, 2002).

A huge number of tools have been developed and improved, with each tool comprising its own specific goals and easements components. Figure 3-1 Classifies the differences between these tools.

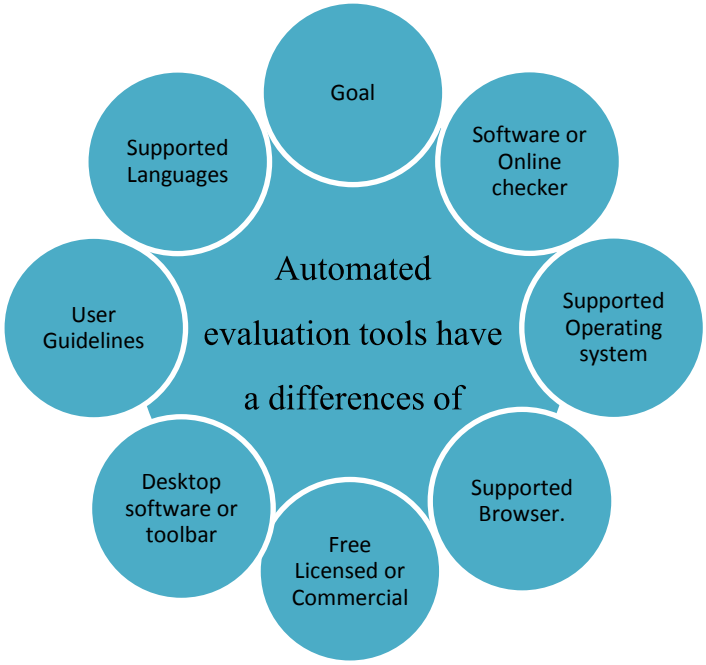


Figure 3-1: Automated evaluation differences

Many different tools are available for the automated evaluation of software. They are very diverse, with each tool offering its own attributes and goals. Examples include WatchFire Bobby, UsableNet LIFT and WAVE, for example; however, automated tools cannot produce some important qualitative information feedback, such as in regard to user preferences. Ivory & Hearst (2001) have identified that only 33% of their 132 reviewed techniques supported automated tools (Ivory & Hearst, 2001). Furthermore, automated evaluation tools are used and still cannot check all the WCAG checkpoints because either they are not covered or

otherwise they are needed for multiple-testing to get the right results (Petrie & Bevan, 2009). This section shows a number of the common evaluation tools still involved in the literature review.

WatchFire Bobby Tool

Bobby tool is web accessibility desktop testing, and was released in 1995, with Version 5.3 released in 2005. The Bobby tool is evaluated in regard to all web elements, including images, audio, and video, so on. The evaluation process relies on Web Accessibility Initiative (WAI) and section 508 Standards. The Microsoft Operating System is the only OS, and is a commercial product (Watchfire, 2005) See Figure 3-2. In 2007 Watchfire Booby become one of the IBM products and become free service (coggnac.com, 2016). The Bobby tool produces a report that broken down into three parts (Thatcher, 2011a):

- Status: Summary of the report that presents either the approval icon if there is no problems found or repair icon if there is any problem found.
- Annotated page: which provide Bobby hat icon for non-compliance issue, and question icon for identified problems which required checking.
- Accessibility errors and questions: presents list of problems with links for manual checking and solving.

UsableNet LIFT Tool

LIFT is developed to evaluate websites and determine problems. LIFT has been released by UsableNet, with many different versions available depending on the developer or needs of the test. LEIFT online is one of the versions that has been developed and released in 2000 for evaluating the accessibility and usability testing solution. LEFT online is able to works on most operating systems, such as Windows, and MacOS (Thatcher, 2011b). LEFT online resource works with the two common web content guidelines, namely WCAG 1.0 and Section 508 (NASA Administrator, 2014). The full function of LEFT online is commercial, with the trial version not providing full evaluation results. According to Thatcher (2011), the main page of the evaluation result contains a list of problems that have been identified, with each problem presented as a link to expand and get more detail about the easiest way to navigate the problems (Thatcher, 2011b). See Figure 3-3 for LIFE tool main web interface

Figure 3-3: has been removed due to Copyright restrictions.

WAVE Tool

This is a tool that helps developers to make the web more accessible. This tool defines and highlights errors. Moreover, WAVE uses WCAG1.0 and Section 508 guidelines to identify the accessibility issues, where WAVE is one of the free tools that can be used online (WAVE Web Accessibility evaluation tool, n.d.). WAVE report presenting the original page with embedded icons and there are six types of icon presented in the results, each with different meanings (see Figure 3-4):

- Read: Error that needs to be solved.

- Yellow: An alert that needs to be checked.
- Green: Accessibility feature.
- Light blue: Problems with structure, semantic or navigation element.
- Purple: problems with HTML5 and Accessible Rich Internet Applications (WAI-ARIA)
- Black: Contrast Errors

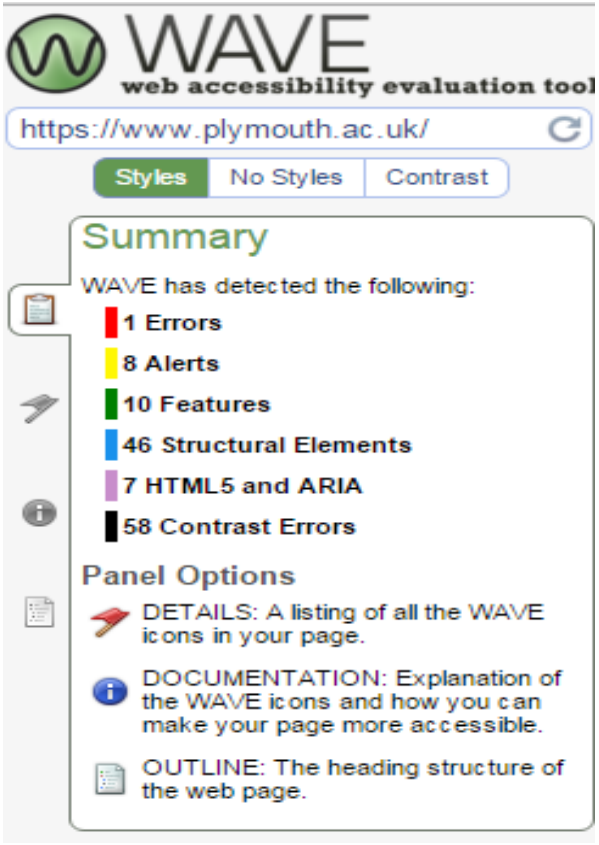


Figure 3-4: Summary of using WAVE tool to test Plymouth University Website

3.2.5 Severity Rating of the Usability Problems

It is important for developers to know about their design problems and the severity level of these problems. Many authors have established a different level of usability severity problems (Dumas & Redish, 1999; Nielsen, 1995; Rubin, Chisnell, & Spool, 2008; Sauro,

2013; Travis, 2009). Most authors have included between three and five levels of severity problems, with each comprising its own description. There are three common usability severity problems.

Nielsen Severity Rating:

Nielsen has established five levels of usability problems (0–4) (Nielsen, 1995):

1. I don't agree that this is a usability problem.
2. A cosmetic problem only: Does not need to be fixed unless extra time is available for the project.
3. Minor usability problem: Fixing this should be assigned low priority.
4. Major usability problem: Important to fix so should be given high priority.
5. Usability catastrophe: Imperative to fix this before product can be released.

Rubin Severity Rating:

This rating have four levels of usability severity problems (Rubin *et al*., 2008):

4. Unusable: The user either is not able to or will not want to use a particular part of the product because of the way that the product has been designed and implemented.
3. Severe: The user will probably use or attempt to use the product, but will be severely limited in his or her ability to do so. The user will have great difficulty in working around the problems
2. Moderate: The user will be able to use the product in most cases, but will have to take some moderate effort in getting around the problem.
1. Irritant: The problem occurs only intermittently, can be circumvented easily, or is dependent on a standard that falls outside the product's boundaries. Also could be a cosmetic problem.

Sauro Severity Rating:

Sauro (2013) defines usability problems under three levels of severity (1–3), followed by user suggestions or comments. The levels are as follows:

1. Minor: Causes some hesitation or slight irritation.
2. Moderate: Causes occasional task failure for some users; causes delays and moderate irritation.
3. Critical: Leads to task failure; causes user extreme irritation.

Insight/Suggestion/Positive: Users mention an idea or observation that does or could enhance the overall experience.

3.3 Summary

In this chapter, there are two main concepts that have been reviewed and discussed. These are design principles and design evaluation methods. Many design principles are established in order to lead developers to create a usable design. However, each developer follows what they believe to be right and comprehensive enough to produce a usable design. On the other hand, lots of evaluation methods also have been established in order to evaluate and assess the design; thus, some methods are classified as inexpensive methods whilst others are classified as expensive. Based on these two main issues, as reviewed in this chapter, there are two main challenges, which are:

- Developers follow different design principles even if these are not comprehensive and do not cover design and evaluation topics.
- A number of developers avoid the conduction of evaluation during the design process.

These challenges have led the researcher to review each issue and devise a list of principles that have been reviewed based on already existing principles. Furthermore, the researcher also has come up with new suggestions relating to the evaluation methods, which are aimed at helping developers to understand and conduct evaluations. This suggestion is related to using the discount usability concept, as established by Nielsen, along with the application of other easy and common methods. Furthermore, various automated evaluation tools can be adopted in an effort to support the use of evaluation methods by software developers. This suggestion is aimed at building a new concept of conducting evaluation methods by developers themselves.

Chapter Four: Software Development Methodologies

This chapter reviews a number of software development methodologies that are applied to create software products. The Waterfall, Spiral and Agile software development methodologies are used in different projects, however the agile approach is widely used this time. The development techniques of User Centred Design (UCD), Scrum and extreme programming (XP) are examined. These development methodologies aim to improve the quality and usability of the software products.

4 System Development Life Cycle (SDLC)

System Development Life Cycle (SDLC) is term used in project management to describe the software phases of the development process from the early planning stage through to the final product release. There are six phases SDLC comprises (Figure 4-1), which are discussed as follows (Alwan, 2015):

1- Planning: to define the issue and how it can be solved.
2- Analysis: to determine system requirements and specifications.
3- Design: to define the system elements, the interface styles, architecture, data types and so on.
4- Implementation: to start coding the requirements and specifications of the system, and accordingly create the software.
5- Testing/Integration: to combine all system components as a united block, before making it ready to test, and subsequently gathering feedback.
6- Maintenance: to fix all errors, and ensure all components are fitting and updated for the latest version.

Figure 4-1: has been removed due to Copyright restrictions.

Many SDLC models have been established and used to create software. Each model has its own specific attributes and goals in application.

The following sections will provide an overview of some of the more common models used, namely Waterfall, Spiral and Agile models. These methods were chosen based on the aim and process of the methods. Furthermore, this chapter reviews a number of common software development methodologies. These methodologies have been reviewed in mind of the following objectives:

- To devise a basic background relating to old and new methodologies and how this is different in order to give new developers a clear view about these methodologies and when these can be applied.
- To identify the similarity (e.g. phases), differences (e.g. liner, iterative, user involving), and advantages and disadvantages of applying the methodologies.
- To show how both old (waterfall) and new (agile) methodologies can be applied, as based on project preferences.
- To show how the new methodologies, such as agile, are more flexible than the waterfall methodology.
- To show how users can be a part of the development process.
- To identify the gap between development frameworks and the involvement of users.
- To identify the gap between development framework and the involvement of evaluation methods.

4.1 Waterfall Model

The waterfall model is considered the classic (and the first) software development methodology, published by Winston W. Royce in 1970 (Sommerville, 2010; Venkataramani, 2014). The waterfall model is the most common software development methodology to have been used, with many software developers adapting it to create their own software processes (Mohammed, Munassar, Govardhan, & Pradesh, 2010). This model includes five phases: requirements, design, implementations, testing and maintenance (Bassil, 2012). Figure 4-2 shows the five phases and the application process of the waterfall model.

Figure 4-2: has been removed due to Copyright restrictions.

These five phased have been described by many authors (Bassil, 2012; Sommerville, 2010).
Requirements Definition: Throughout this phase, all requirements should be determined, including a comprehensive analysis for both functional and non-functional requirements. The

hardware that will execute the program and the software development language will also be determined.

System and Software Design: In this phase, the collected user requirements are used to design a detailed software solution that meets all defined user requirements. Extensive use is often made of the UML specification to create diagrams identifying the software's components and the relationships between them (examples include Use Case diagrams, Sequence diagrams and Class diagrams). Software architects anticipate every problem that could arise and provide the design with a solution. In practice, this is extremely difficult to do, primarily because, unless the problem the software aims at solving is very simple or very well understood, it is impossible to anticipate all problems that could arise. It also relies on the completion of the requirements gathered, which is unlikely for a complex problem.

Implementation and Unit Testing: In this phase, the software is built (coded) based on the collected requirements. Most software is modular, with each module implemented as a separate unit. Each unit will be tested individually in order to determine whether or not the user requirements have been met.

Integration and System Testing: In this phase, all software modules are combined with the other modules to create the finished software solution. At this time, the system is tested as one complete program, and its performance is validated against the user requirement specification. It is important identify and rectify any errors in the software at this point.

Operation and Maintenance: In this phase, the system is deployed and operational use begins. The users will identify areas requiring improvement or modification, which will lead to maintenance work being performed of the software and, in the long-term, possibly new requirements. Eventually, there will be the need to re-design some portion (or all) of the software.

The waterfall model has no overlap between its phases, and each phase needs to be completed before moving on to the next phase. For this reason, many software developers understand that the waterfall model cannot be applied to all types of software project: it is most suited to small, well understood problems, where the requirements can be clearly defined and do not usually change (Sommerville, 2010).

The following advantages and disadvantages have been identified for the waterfall model (Pressman, 2010; Ruchika, 2012; Seema & Malhotra, 2012):

Advantages of the Waterfall Model:

1. Easy to apply

2. Commonly used in industry
3. Enhances good habits, 'define before design, design before code'
4. Suitable for a well understood problem that needs to be solved by a software product, and development with a weak team
5. Documents are produced in the early stages, which make the project easy to understand.

Disadvantages of Waterfall Model:

1. No way of getting back to the previous phase
2. Deployed in a 'big bang' approach at the end of the process, where incremental deployment does not occur
3. Small errors in the final product may produce a lot of problems up to and including changing major design choices
4. It is impossible to expect all requirements to be identified in the early stages of the project, where the only exception is a very well understood problem (for example: a Tic-Tac-Toe game)
5. It is difficult to integrate risk management.

The best type of software project to employ the waterfall methodology is one where (Seema & Malhotra, 2012):

- Project requirements are easily identifiable
- Project requirements are unlikely to change
- The problem being solved is very well understood (effectively, where a standard solution already exists).

4.2 Spiral Model

The Spiral Model is a software framework used to create software products. The Spiral Model was established in 1988 by Boehm (Sommerville, 2010). Unlike the waterfall model, which relies on performing a sequence of operations, the Spiral Model uses iterative development to produce ever more accurate 'prototypes' of the software. Eventually, a prototype will be developed that meets all user requirements. Essentially, the same steps of development are 'iterated over' to make each prototype. This model encompasses the following four phases (Mohammed *et al*., 2010);

- Determine objectives,
- Identify and resolve risks
- Development and testing
- Plan the next iteration.

Figure 4-3 shows the spiral model phases in more detail.

Figure 4-3: has been removed due to Copyright restrictions.

Each loop in the spiral model produces a part of the software, where each loop is divided into four sections. These are (Mohammed et al., 2010; Sommerville, 2010):

1. Objective-setting: The phase's objectives should determine all design constraints and alternative design strategies that could be applied.
2. Risk analysis: Risks that may prevent or interfere with the production of the next prototype should be assessed. The actions that can be taken to reduce these risks are identified and plans to implement them are prepared.
3. Development and validations: The prototype should be designed, developed and then tested to prove it meets requirements.
4. Planning: The prototype is reviewed and the next iteration is planned and requirements are gathered.

The Spiral Model, like any other development methodology, has strengths and weaknesses when applied in practice.

Advantages of the Spiral Model:

1. Flexible
2. Easy to observe project progress
3. Considers risk analysis at each iteration of the project and offers the opportunity to abort
4. Appropriate for large and high-risk projects (for example: research projects attempting to do something that has never been done before)

5. The software appears in the early stage of the process, which offers many advantages from the early testing of the product to the opportunity to obtain early user feedback.

Disadvantages of the Spiral Model:

1. A high-cost development methodology with the danger of generating cost overruns if not controlled well.
2. No 'natural' end to the spiral so management must determine when the product is 'good enough'. Further development wastes resources and money.
3. Expert evaluators are required for the frequent review of the project.
4. Not appropriate for small and low-risk projects.
5. The project completion criteria depend on the risk analysis stage. Often, in the real world, the costs/benefit analysis of continuing to develop prototypes is not easy to correctly determine.
6. Expert risk analysis is needed.
7. The production of huge documentations makes project management too complex (Mohammed et al., 2010; Sparrow, 2012).

Applying the Spiral Model may be suitable if the software project meets the following criteria (Seema & Malhotra, 2012):

- Creating a prototype is required or seen as the best way of proceeding (such as when you are not sure that it is even possible to complete the project)
- The project is high-risk, and the assessment of the risk is important (for example: a research project where the outcome can be a success or failure)
- The project requirements are complex or cannot be well-defined
- Changes to user requirements are expected to occur all the time
- The project is a long-term one, such as developing an operating system.

4.3 Agile

Agile software development (ASD) is 'a methodology for the creative process that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product. Agile software development focuses on keeping code simple, testing often, and delivering functional bits of the application as soon as they're ready' (Rouse, 2007). Agile is

based on the iterative development concept and aims at minimising the time spent designing the solution (see Figure 4-4) (Eklund & Levingston, 2008).

Figure 4-4: has been removed due to Copyright restrictions.

Beck *et al*. (2001) have listed the 12 principles behind the agile development, which are (Beck *et al*., 2001):

1. The highest priority is to satisfy the customer through the early and continuous delivery of valuable software.
2. Welcomes changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from every few weeks to every few months, with a preference for the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build the projects around motivated individuals. Give them the environment and support they need, then trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is a face to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximising the amount of work not done is essential.
11. The best architectures, requirements and designs emerge from self-organising teams.
12. At regular intervals, the team reflects on how to become more effective, and then readjusts its behaviour accordingly.

In 2008, the Vision One survey was conducted to identify the benefits behind agile methodology approaches. Developers who are already employing the agile methodology were asked to rate a number of statements on the five-point Liker scale. Figure 4-5 shows the percentage of the study participants who rate the statement as 'significantly improved' or 'improved' for each suggested statement (VersiononeSurvey, 2008).

Figure 4-5: has been removed due to Copyright restrictions.

Nowadays, the agile software development methodology has led to the creation of many software development techniques that support the agile concept: for instance, SCRUM and eXtreme Programming (XP) (Extremeprogramming, 2013; Scrum, 2015).

4.3.1 SCRUM

Figure 4-6: has been removed due to Copyright restrictions.

Scrum is 'a management and control process that cuts through complexity to focus on building software that meets business needs' See Figure 4-6 (Scrum.org, 2015). Scrum is one of the famous agile techniques used to manage and control the software building process, and has been used since the 1990s (Schwaber & Sutherland, 2013). In scrum development, there are three main parties (James, 2010):

- The Product Owner: Responsible for long-term project vision. Reviews the product and ensures a positive Return on Investment (ROI). Often serves as the 'problem domain' expert on the development team. This role is generally performed by a business or customer representative.
- The Development Team: All the development team members working as self-organising to achieve their tasks. Members of the team are not limited to developers; they would include problem domain experts and specialists in analysis (mathematicians/statistics) and design (software testers). The team is usually between 5 and 9 members in size.
- The Scrum Master: The Scrum Master's main role is as a facilitator who ensures that the scrum process happens and removes impediments to the scrum team's progress. Importantly, the scrum master does not manage the team; if a person has managerial authority over the development team, they are excluded from being the scrum master.

Using "Kanban board" it's helpful to perform by the Scrum master to monitor the sprints process and determine which tasks have completely achieved and which don't.(see Figure 4-7)

Figure 4-7: has been removed due to Copyright restrictions.

The Project Backlog

The Project Backlog is a prioritised list of features, usually grouped together into potentially shippable product groups.

It may contain:

1. Features
2. Bugs—'effectively new features'
3. Technical work
4. Knowledge acquisition.

First, sprint should almost always be creating the product backlog as all other sprints will begin by selecting entries on the product backlog for inclusion in the 'sprint backlog'. Most entries in the product backlog will adopt the form of a 'User Story'. Cohn has defined user stories as ' short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system'(Cohn, 2015c). User stories are composed of three main components (3Cs):

1. Card
2. Conversation
3. Confirmation.

Sprint Backlog

This backlog is created before each and every sprint. Its content is agreed between the product owner and the sprint team. It is a list of features (usually expressed as user stories) which, when implemented, will fit a potentially shippable product increment. Once a feature is moved to the current sprint backlog, its requirements are 'frozen' and cannot change until the end of that sprint. The first task of the sprint planning meeting is to agree the sprint backlog. Once the sprint is planned, work begins and is monitored through the daily scrum

meeting. A sprint is a 'timebox' with a fixed amount of time to complete the sprint backlog. At the end of the sprint, the sprint review meeting is held

The scrum process involves five key events (Cohn, 2015a; Schwaber & Sutherland, 2013; Wall-Skills.com, 2014):

❖ **Sprints:**

- Time divided into sprints
- Each sprint produces a potentially deliverable product
- Each sprint takes 1–4 weeks.

❖ **Sprint Planning**

- Plan for each sprint with developers and the product owner to find what they can do for the sprint
- 8 hours of planning time allocated to every 4 weeks
- Review the items selected for development in the sprint
- Discuss the time required to finish the sprint items and distribute time/work to scrum team members
- Scrum Master is responsible for ensuring the planning meeting happens.

❖ **Daily Scrum Meeting**

- No longer than 15 minutes
  - Each team member reports:
    - What they achieved yesterday (progress made)
    - What they expect to work on today
    - What they expect to work on tomorrow.

❖ **Sprint Review**

- No longer than 4 hours for each 4-weeks sprint
- Product owner reviews the teamwork that should have produced a potentially shippable product increment
- The product owner will accept/reject the potentially shippable product increment
- Participants in the sprint review typically include 'the product owner, the Scrum team, the Scrum Master, management, customers and developers from other projects' (Cohn, 2015b).

❖ **Sprint Retrospective**

- 3 hours for each 4-week sprint
- Aim at improving the scrum process for each sprint

- Lean from the previous sprint mistakes
- Often conducted on the Start-Stop-Continue approach where participants identify things that in future they will:
    - Start doing,
    - Stop doing, and
    - Continue doing.

4.3.2 eXtreme Programming (XP)

Extreme Programming (XP) is a software development methodology that adopted an agile concept. XP was applied by the Kent Beck (1996) project, which he defined as 'a lightweight methodology for small-to-medium-sized teams developing software in the face of vague or rapidly changing requirements' (Beck & Andres, 2004). XP aims at bringing all the development team to work together and is recommend for use if the development team provides a small number of members (Extremeprogramming.org, 2009). Beck & Andres (2004) have listed many advantages of using XP methodology; however, Extremeprogramming.org (2009) have classified these advantages into five categorised, as follows:

1. Simplicity: This means the developer will do what they can do and what is required to do only to come up with simple solution.
2. Communication: This means daily face-to-face communication between the development team is an essential value behind the application of XP. This communication spans from the beginning of the design process until the final completed version of the solution; thus, the solution will be created by all team members.
3. Feedback: This means XP considers the communication of feedback as an essential value to improving the software; thus, the development is based on the regular meeting feedback of team members.
4. Respect: This means everyone on the development team will ensure full respect of their own work, even if it is a small task. Therefore, XP is an integral work.
5. Courage: This means sharing knowledge, experience and design progress between all members is an essential way of coming up with a good solution. Thus, in XP, nobody works alone, and everyone should know what's going on.

Twelve practices have been listed behind the use of the XP methodology (Ronjeffries.com, 2011):

1 Planning game
2 Small releases
3 System metaphor
4 Simple design
5 Continues testing
6 Refactoring
7 Pair programming
8 Collective ownership
9 Continuous integration
10 40-hour work week
11 On-site customers and
12 Coding standards (Ronjeffries.com, 2011).

Beck & Andres (2004) classify the core practices into two categories See Table 4-1

Table 4-1: has been removed due to Copyright restrictions.

Extreme Programming can be conserved as a collection of continuously occurring feedback loops that apply the above principles to create and release computer software (code). Figure 4-8 shows the feedback loops of extreme programming.

Figure 4-8: has been removed due to Copyright restrictions.

Each feedback loop 'happens' on a different timescale, ranging from seconds in a pair-programing environment to months for an overall software release plan. Throughout this time period, each loop is continuously 'running' to generate the software project's code base.

4.4 User-Centred Design (UCD)

User-centred design (UCD) is a way of improving overall software usability (Bowler et al., 2011; Jokela et al., 2003). User-centred design is a well-established software development methodology that incorporates design evaluation within the core development lifecycle (Bowler *et al*., 2011) . There are different terms used in the field that have the same concept of UCD: for instance User-Centred System Design (UCSD), Usability Engineering (Rannikko, 2011) and Human-Centred Design (HCD) (M. Maguire, 2001). The term UCSD originates from research by Norman (Vredenburg, Smith, Carey, & Mao, 2002) back in the 1980s. Since then, UCSD has been widely adopted (Abras, Maloney-Krichmar, & Preece, 2004; Keinonen, 2010). It focuses on user's requirements and needs during the software development stages (Preece et al., 2002) . The concept of UCD emphasises the involvement of the user and their activities in order to achieve product goals.

4.4.1 The Principles of User Centred Design (UCD)

Since USD was developed, there has been no definitive list detailing its underlying principles, despite various attempts by researchers and official bodies to develop such a list. Each author has devised their own list of principles, with the choice of authors listed and discussed in Table 4-2 (J. D. Gould, Boies, & Ukelson, 1997; J. D. Gould & Lewis, 1985; Gulliksen et al., 2003; ISO 13407, 1999; ISO 9241-210, 2010).

Table 4-2: Summary of UCD Principles

Year Proposed	Researcher/ Standard	Principles
1985	**Gould & Lewis**	1. Early focus on users and tasks 2. Empirical measurement 3. Iterative design
1997	**Gould *et al***	4. Integrated design
1999	**ISO13407**	1. The active involvement of users 2. An appropriate allocation of functions between user and system 3. The iteration of design solutions 4. Multidisciplinary design teams

Year Proposed	Researcher/ Standard	Principles
2003	**Gulliksen**	1. User focus 2. Active user involvement 3. Evolutionary systems development 4. Simple design representations 5. Prototyping 6. Evaluate use in context 7. Explicit and conscious design activities 8. A professional attitude 9. Usability champion 10. Holistic design 11. Processes customization 12. A user-centred attitude should always be established
2010	**ISO 9241-210**	1. The design is based upon an explicit understanding of users, tasks and environments. 2. Users are involved throughout design and development. 3. The design is driven and refined by user-centred evaluation. 4. The process is iterative. 5. The design addresses the whole user experience. 6. The design team includes multidisciplinary skills and perspectives.

Gould & Lewis's Proposed Principles:

In 1985, Gould & Lewis put forward the following four (4) principles as underlying User Centred Design (UCD) (J. D. Gould & Lewis, 1985):

Early focus on users and tasks: In the early stage of the design process, the end users of the product should be identified in relation to their attitudes and characteristics. The developers should meet real system users to assess their needs, attitudes and characteristics, as opposed to reading requirements, documents or hearing about requirements from another party. Real face-to-face interviews and discussion sessions are a better way of collecting user requirements. Furthermore, the tasks which users are expected to perform should be discussed based on user knowledge and abilities.

It is expected that, by involving users early on in the design process, the overall development time and costs will be reduced. The final design will be more likely to meet user requirements and pass acceptance testing if the user defines these with the development team at the outset.

Empirical measurement: Both learnability and usability measurements should be considered during the design process so as to ensure the product will be easy to use. Early testing identifies problems and allows remedial work to occur before the system grows too dependent on flawed implementation code. This can be achieved by making a simulation or prototype of the system and accordingly observing user reactions. Experimental studies, for example, measure the memory access speed, whereas empirical studies simply measure speed but do not assess whether or not the speed is acceptable to the system users.

Developers and software engineers focus on functional requirements that can be empirically measured; however, users also judge a system on non-functional requirements. These are often difficult to identify and therefore are often omitted unless a user evaluates early models of the system. Early evaluation and identification of these requirements prevents the wasteful redesign of the system at a later stage.

Iterative design: The ability of the redesign of the product is essential to consider, with focus on the design process. Essentially, the design process should permit a redesign of the products components as additional user requirements are identified or existing ones change. Product evaluations produce a list of errors, and problems should be fixed with the design modified to correct for these errors. A process of iterative design enables this to occur.

The iterative design approach is helpful in terms of improving design by quickly including the evaluation results. Furthermore, iterative design allows changes in the system requirements to influence the design of the system.

In 1997, Gould updated his principles and added a new principle, namely integrated design. In this case, the various components, namely online help, the system overall, training, organisation and work practices need to be developed in unison, with one common objective directing development.

International Standards Organisation and ISO 13407:

Two years later, the International Standards Organisation put forward their own definition for UCD's fundamental principles in the form of ISO 13407. The ISO adopted some of Gould's principles, specifically (1) and (3), but also included the following two that Gould had not suggested:

An appropriate allocation of functions between the user and system: when establishing the allocation of functions between human and computer system, the following guidance was provided by (Liu, Zuo, & Zhang, 2011):

1. Human and machine comparison: Compare the abilities of humans and computers, and accordingly allocate the functions in an objective and logical way.
2. Cost focus: Allocate the functions of humans and machines according to the comparison of cost and benefit (Dearden, 2000).
3. Human-centeredness: Improve the overall satisfaction of users as much as possible when allocating functions of information systems.

The key point here is that the computer's ability does not always drive the allocation of functions. Point 3 advises that human satisfaction is of overriding concern so, for example, whilst the 'self-driving car' is a technical possibility, it remains to be seen whether the user will be satisfied with giving up car control.

Multidisciplinary design teams: Each design is collaborative processes involving many people to complete the design. The UCD should contain a team with different expertise, such as manager, evaluation expert, graphical designer end users, etc. (Jokela *et al*., 2003).

ISO 13407 partly depends on Gould's principles and accordingly builds upon them to create the main definition of UCD's principles. The first added principle of 'function allocation' between computer and user emphasise that the abilities and preferences of users cannot be ignored. The second added principle is focused on design team management and the need for a team with multiple skills relevant to the problem that the design should solve. The ISO, at this time, has covered more areas than Gould's principles.

Gulliksen's Proposed Principles of UCD:

In 2003, Gulliksen *et al*. proposed a much more detailed set of principles for UCD. They came up with 12 principles, drawing on Gould's principles, the ISO13407 standard and existing best practise. These 12 principles replaced Gulliksen's first attempt at defining UCD principles in 2001 (Gulliksen & Göransson, 2001). Despite the revision, their 12 principles seem to lack focus concerning the difference between a fundamental principle and how it should be applied.

Table **4-3** presents the 12 principles and relates them to the principles of ISO 13407 and Gould. This table shows that the first and second principles are restatements of Gould's and the ISO13407 standard's first principle. The third and fifth principles are repetitions of Gould's and the ISO13407 standard's third principle. Moreover, the sixth principle is identical to Gould's second, and the seventh principle is a restatement of the ISO13407 standards second principle. The tenth principle is a repetition of Gould's fourth principle. Both the eighth and ninth principles are restatements of ISO13407 standard's fourth principle. The remaining fourth, eleventh and twelfth principles seem to be recommendations on how the UCD principles should be applied as opposed to fundamental principles in their own right.

In summary, not all of these 12 principles are fundamental principles that are required for UCD; rather, these re-state principles in a different way or otherwise provide guidance on how to apply them. For example, the fifth principle requires the use of prototypes whilst the third requires evolutionary development—a series of ever better prototypes IS evolutionary development. Hence, essentially, these two are the same thing. Thus, these principles are good as a reference to understanding the ISO 13407 and Gould's principles of UCD.

In 2010, ISO9241-210 updated their principles but did not include the majority of these 12 principles. Clearly, then, the ISO was not convinced that these where fundamental UCD principles.

Table 4-3 Contrasting Gulliksen's with the ISO 13407 standard and Gould's UCD principles.

Gulliksen's	ISO 13407	Gould's
1	1	1
2	1	1
3	3	3
4	X	X
5	3	3
6	X	2
7	2	X
8	4	X

9	4	X
10	X	4
11	X	X
12	X	X

International Standards Organisation and ISO 9241-210:

In 2010, after twelve years, the ISO updated its definition of the UCD principles by releasing a new standard—ISO 9241-210. The total number of principles grew from 4 to 6. The only completely new 'principle' was the requirement to include user-centred evaluation of the design. The fifth principle requiring 'the design addresses the whole user experience' expands on the definition of 'easy to do' so that it is not limited to only being easy to perform a function. The expanded definition also includes the idea of considering the emotional and perceptual aspects of users' experience, as well as how 'easy' it is to perform a function (Travis, 2011).

4.4.2 The Benefits of UCD

Creating usable software is the main overall benefit of applying the UCD approach in the development lifecycle. Bevan (2005) lists a number of benefits behind the UCD main concept (see

Table 4-4) (Bevan, 2005).

Table 4-4: has been removed due to Copyright restrictions.

4.4.3 UCD Activities:

ISO states that there are four activities of developing design (see Figure 4-9) (ISO 13407, 1999; ISO 9241-210, 2010). These activities have been described in the following (M. Maguire, 2001; Teoh, 2006):

1- Understand and specify the context of use: The purpose and goal of the application under development should be clearly identified. Users' interactions and the purpose of those interactions need be clear to developers.
2- Specify user organisational requirements: The requirements collection is the important phase of the software development process. Both user and organisation requirements of the software need to be identified at an early stage of the development process. Many methods can be applied to collect the requirements—for instance: interviews, focus groups, scenario of use and so on. The design specification and functions also need to be specified before creating the design.
3- Produce design solutions to meet user requirements: There is no specific development approach required that should be applied to create the design; however, the design process needs to be an iterative one. Several techniques can be used to support iterative design, such as mock ups and simulations of the system, for example. These two techniques help to produce the system at an early stage of the development process; thus, developers are able to evaluate the system with real users. The design creation techniques, guidelines and prototypes are important in creating usable software.
4- Evaluate design against requirements: The evaluation, by users, against the original requirement, is an essential activity in the development process. Overall progress should be measured in terms of user requirements that have been met. The evaluation should start at early stages of the design process. A number of evaluation methods can be applied by both end users and domain experts to determine the requirements that have been met. The evaluation sessions results lead to design improvements that incorporate many different points view, such as the software's overall effectiveness and efficiency, and users' satisfaction.

Figure 4-9: has been removed due to Copyright restrictions.

4.5 Summary

This chapter has provided an overview of a number of common software development methodologies. This review has shown how these methodologies are fitting on different projects, which depend on the aim of the project and its length, as well as how the main goal of the methodology reviewed has identified the similarity and differences between methodologies in terms of phases and the process. What is missing, however, is a tool that incorporate each phase in order to give developers a list of evaluation methods so as to improve software usability. Thus, each phase of the development process and the moving process between phases is important in devising a usable product. At the stage of review, the researcher comes up with a general concept tool to promote novice developers in the application of methods during the development process. This suggestion tool includes a number of methods; thus, it can be incorporated with each development process phase and the developer can choose and apply the appropriate method. There are two purposes behind our suggested tool: firstly, to increase the use of evaluation methods and improve the overall usability; and second, to let novice developers improve their knowledge surrounding evaluation methods, alongside technical skills, and improve their behaviour of developing products. It is valuable to combine between methodologies concepts in order to improve the usability. For instance, in regards the combination between agile and UCD, the agile concept leads the iterative and UCD concept, which leads to user involvement on the development

process. Thus, this combination will integrate the two concepts into one tool; however, how to create the tool, how to promote novice developers to use the tool, and how this tool changes novice developers' designing behaviours are the challenges.

The next chapter will show how the combination between development methodologies can improve design usability.

Chapter Five: Integrating Agile with UCD towards A Theory of Integrated Development

Software development is currently part of industry and research. Many different methodologies have evolved over the years. However, these days, agile software development is one of the key software development methodologies used. Thus, this chapter provides a practice of integration Agile and UCD. It's important to know about the integration ability, proposed principles, the integration benefits and challenges. Finally, the chapter is summed-up with our suggested development approach.

5 Integrating Agile with UCD towards A Theory of Integrated Development

5.1 Introduction

In 2001, a number of developers had been met to establish the manifesto for agile software development. They then came up with 4 values (Beck et al., 2001):

1. Individuals and interactions over processes and tools
2. Working software over com behavior prehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan.

Based on these four values, they devised 12 principals in mind of helping the use of the agile concept. These 12 principles are mentioned in chapter 4 Section 4.3 (Agile section). User-Centred Design (UCD) is established in an effort to put the user at the centre of the design development process; emphasis is placed on the user's requirements and needs during the software development stages (Preece *et al*., 2002). UCD is widely adopted with other development methodologies in order to increase design usability (Abras *et al*., 2004; Keinonen, 2010). The integration of agile and UCD is one of the most important integrations centred on improving design usability goals (Fox, Sillito & Maurer, 2008; Humayoun, Dubinsky & Catarci, 2011; Marc, 2013; Najafi & Toyoshiba, 2008; Salah, 2011; Sharp, Robinson & Segal, 2004).

In the following section, four main sections will be presented, namely the best practice, positional benefits, principles, and challenges of integration towards the new approach of integration.

5.2 Best Practice

More recently agile software development (SD) has been integrated with other development processes: for instance, User-Centred Design (UCD). This integration aims to improve the level of software usability by combining the strength of both approaches and integrates them in one model to solve development issues and challenges. For instance, user involvement is one of the challenges facing developers during the development process; thus, integration of

UCD is a way of addressing this challenge. Many authors have proposed different integration frameworks for different levels of integration, where each framework has its own goals (Fox, Sillito, & Maurer, 2008; Humayoun, Dubinsky, & Catarci, 2011; Marc, 2013; Najafi & Toyoshiba, 2008; Salah, 2011; Sharp, Robinson, & Segal, 2004). The most relevant of these frameworks is discussed further in this chapter.

Additionally, Hussain *et al*. (2009) conducted a study to investigate the current state of the integration of agile and UCD. This study involved 92 participants, all of who already work on the integration of agile and UCD as usability professionals or software developers. The results of the study show that most of the participants rate this integration as significant and having added value for the design process and the design teams. It was also found to have an impact on product improvement, quality, usability and user satisfaction (Hussain, Slany & Holzinger, 2009). Thus, the integration of usability evaluation methods and agile software is an essential development approach to delivering a usable product by applying more iterative and testing during the process (Sohaib & Khan, 2010). However, this integration is still not employed in a large number of companies (Silva, Silveira & Maurer, 2015).

5.3 Integration Potential

Since both agile SD and UCD are iterative approaches, it is important to establish whether these two approaches empirically work together. Fox *et al*. (2008) researched the effectiveness of integrating agile SD with UCD. This study involved participants who already adopted a combination of these concepts. This study reports evidence which suggests that agile and UCD work well together. When they are presented in one model (Fox et al., 2008)(Sohaib & Khan, 2010). Thus, the integration of UCD with agile increases of software developed usability (Zahid Hussain, Slany, & Holzinger, 2009)

Additionally, using evaluation methods throughout the design development process is valuable for the level of quality and usability of the design. Usability evaluation methods are used to identify user requirements and accordingly gather feedback. Thus, Najafi & Toyoshiba (2008) found that evaluation methods that are incorporated with in the agile SD process can be a way of increasing the overall usability of the final product. Furthermore, this integration does not impact the developing process schedule (Najafi & Toyoshiba, 2008). Furthermore, Marc (2013) found that end users should still evaluate the software; even if the

whole product is incomplete. Moreover, the combination of agile and UCD enables the development team to regularly involve end users in the design process (Marc, 2013).

Sohaib and Khan (2010) completed a comprehensive study of agile SD and usability engineering topics, exploring the strengths of each concept. This study recommends an approach adopted in agile and usability engineering concepts:

i. *Iterative development throughout the project*
ii. *Assemble a multidisciplinary team to ensure complete expertise*
iii. *Collaboration between customers, users, product managers, business analysts, developers will maximise overall team efficiency for usable products*
iv. *Unit Testing, User Acceptance Testing and Usability Testing throughout the process* (Sohaib & Khan, 2010)**.**

5.4 Integration Benefits

Integration brings many benefits that impact the final product design. Marc (2013) states that integration with UCD has a number of benefits (Marc, 2013):

1. *Better understanding of the problem*
2. *Allows rapid testing and validation of story concepts before time consuming coding*
3. *Provides a clear, sociable visual representation of the project vision*
4. *Provides usability by stealth*
5. *Engaging the end user as a customer*
6. *Improves basis for estimation* of the final application design, the functionality and the estimated effort for the implementation
7. *Mitigates project risk.*

Eklund & Levingston (2008) make two recommendations for developers who are interested in integrating usability testing with agile development. Firstly, user involvement should be a core part of the development process; and secondly, expert evaluators' involvement is a way of reviewing the design in the early stages of the development process, also in order to consult the time and scope of the testing. However, the involvement of both users and experts does not need to be formal; Informal involvement works well (Eklund & Levingston, 2008).

The above-mentioned scholars propose that expert evaluators need to be involved in the development process in order for agile SD to improve in terms of design usability. Furthermore, involving experts does not need to be formal or well-planned. Informal

reviewing via short meetings or emails has also proven to be useful for solving at least the simple problems (Eklund & Levingston, 2008).

5.5 Integration Principles

Chamberlain *et al*. (2006) proposed a framework that integrates agile with UCD for use by SD project teams. This study established five principles that are recommended for the integration of agile SD and UCD. The first principle centres on user involvement in the development process. Secondly, both designers (who responsible for user-centred activities) and developers (who created the code) should closely work with each other. Thirdly, designers are required to 'feed the developers' with system prototypes and user feedback. Fourth, UCD practitioners must be given plenty of time in order to learn the basic needs of end-users before any coding takes place. Finally, the integration of agile and UCD have to be within a cohesive project management framework (Chamberlain, Sharp & Maiden, 2006).

Eklund & Levingston (2008) have devised four recommendations that help agile SD developers to get sound feedback on integrating usability testing into the agile development process. These are:

1. The development team should fully understand the concept of user-centred design of involving users at different stages of the development process
2. Expert evaluator involvement is essential to solve the design problems and speed the design process
3. Many smaller testing rounds are better than one big testing round
4. A new collection of requirements is needed. Thus, usability testing is a way of collecting new requirements and reporting these to developers as soon as possible (Eklund & Levingston, 2008).

In 2015, Brhel *et al*. reviewed the literature of agile and UCD methodologies to identify the generic principles establishing user-centred agile software development (UCASD). This reviewed study identified five principles for agile SD and UCD integration (UCASD), namely (Brhel, Meth, Maedche & Werder, 2015). There are: (1) Separate product discovery and product creation phases. (2) Use iterative and incremental design and development. (3)

The parallel interwoven creation tracks. (4) Keep continuous stakeholder involvement, and (5) Used artifact-mediated communication to document both product and design concepts.

5.6 Integration challenges

Any integration has potential to face challenges, which researches attempt to identify in order to resolve them. Chamberlain *et al*. (2006) mention power struggles and time differences capacities between developers and designers. Moreover, team miscommunication and unwillingness to learn users' needs and accept user involvement in the project significant impacts on the effectives of the integration (Chamberlain *et al*., 2006). Furthermore, the lack of collaboration between developers and designers is also confirmed as a challenge for integration as designers work on numerous projects at the same time, meaning they are busy (Silva da Silva, Selbach Silveira, Maurer & Hellmann, 2012). Thus, project management planning should be created to resolve this issue.

In 2014, Salah *et al*. published a review of the existing literature on the agile SD and UCD from the year 2000 through to 2012. The aim of this work was to identify the fundamental challenges for the integration of agile and UCD. As a result, seven challenges were found to impact the integration of Agile SD and User Centred Design Integration (AUCDI). These challenges were in UCD infrastructure, people, and process: (1) Lack of Time for Upfront Activities; (2) Difficulty of Modularisation/Chunking; (3) Difficulty of Prioritising UCD Activities; (4) Optimising the Work Dynamics Between Developers and UCD Practitioners; (5) Performing Usability Testing; (6) UCD Practitioner Workload; and (7) Lack of Documentation (Salah, Paige & Cairns, 2014).

Cheap and quick evaluation methods are also attempted to integrate with agile SD. Discount usability is concept of using cheap and quick evaluation methods that defined by (Nielsen, 1994). However, discount usability has weak integration with agile development; this issue is considered a challenge of integration. Thus, this challenge needs to be explored and resolved so as to improve the level of software usability and productivity (Eklund & Levingston, 2008; Kane, 2003). Sohaib and Khan (2011) proposed a framework that integrates the agile SD (XP) with discount usability methods. In their framework they adopt each method in specific phases of the development process. The integration was achieved as specified below:

1. Scenarios and user stories (Exploration phase)
2. Card sorting (Planning phase)
3. Heuristic evaluation (Acceptance testing)

4. Thinking aloud (Productionizing phase) (Sohaib & Khan, 2011).

Silva da Silva *et al (2004)* found that integration needs for more empirical studies regarding the integration of UCD and agile SD methods(Silva da Silva, Martin, Maurer, & Silveira, 2011). Furthermore, Sharp *et al*. (2004) believe that universities are the best environment for promoting the idea of creating integration between agile SD and UCD as an education discipline will impact the industry when students work and understand of UCD and HCI concepts (Sharp *et al*., 2004).

5.7 Summary:

This chapter shows that the integration of software development methods with real users is possible and can be applied. Thus, integrating agile with UCD is a focus of many researchers, where each research provided different contributions. These researches are recognised as the main key for creating the main concept of the new framework as it is a background for a number of practices. Moreover, this chapter shows evidence to support that the integration of agile and users can still be improved so as to create usable products. Furthermore, the list of recommendations for these studies also was clear, and shows that integration helps practitioners to consider two main element of integration: first, the iterative design; and second, user involvement. Based on the results of the current studies, a new framework will be created so as to enhance the use of evaluation within the development process. The number of challenges and benefits can be considered on the new framework concept. For instance, miscommunication between developers and designers is one challenge of integration; thus, a reduction in the number of project teams, especially for small or student projects, is important. Furthermore, one person is expected to create the design and evaluate in mind of all management issues. Our suggested framework is considered to be one of the integration practices attempting to integrate agile (iterative approach concept), user involvement (UCD concept) and, above them, evaluation methods.

Chapter Six: Research Methodology

This chapter provides a research philosophy, a discussion on the various data collection and analysis methods applied in the present work. Mix methods including number of data collection methods were carried out in a bid to garner empirical data. Convenience sampling has chosen in purpose to determine the study participants.

6 Research Methodology

6.1 Introduction

The mind-set and perceptions of novice developers needs to be improved in regards usability; this in turn will result in greater approaches to assessing software usability. Such improvement can be achieved through establishing new instruments with good knowledge; this will assist the novice in the successful running of usability evaluation sessions. Creating such a tool requires that a good understanding be held by researchers in terms of the most important factors in improving developer mind-sets. In line with the researcher's own resume, the majority of the improvement needs are experienced, which well positions the researcher in creating such a tool. Thus, this chapter is comprised to consider the most appropriate methodology, and the devising of such, in mind of providing answers to the research questions, and accordingly satisfying the outlined research objectives. The research philosophy, strategy design, and methods selected for the work will be discussed in this chapter, in addition to the justifications behind their selection. Moreover, it highlights the quantitative and qualitative data collection and analysis methods adopted in the work, and the reasons for such choices. However, the chapter does not consider the specific use of such data collection and analysis methods, with this area tackled in the subsequent chapters.

6.2 Research Aims, Objectives and Questions

The current work focused on improving general awareness and assessment practices of novice developers in the implementation of usability evaluation methods. It is the overall nature of this research that is valuable when considering that there has been a lack of studies carried out on this topic, analysing the potential of novice software engineers to complete usability assessments for their products. It is this gap in the literature that provides a motivational factor behind the completion of this study.

This research applies both quantitative and qualitative methods, with Stenius *et al*. (2008) stating that 'The combination of qualitative and quantitative methods can deepen the understanding of processes, attitudes, and motives' (Stenius, Mäkelä & Miovsky, 2008). Such a mix is also referred to as mixed-method approaches, which infers the use of both

quantitative and qualitative data collection methods and analyses procedures (Saunders, Lewis & Thornhill, 2009). Multi-case study strategy has been adopted in an effort to understand and further establish the various research aspects. Quantitative research seeks to highlight what is happening, whereas qualitative research, in contrast, seeks to provide understanding as to why this is happening. The goal is centred on achieving greater levels of understanding amongst individuals in regards their attitudes, behaviours and views (Moore, 2006).

The quantitative analysis aspect is focused on the questionnaire responses. The population of the sample comprised novice software developers of both genders from Plymouth University. The sample underwent a quantitative analysis, with the developers well positioned to determine and further develop more in-depth insight into all factors. Accordingly, the use of semi-structured interviews was considered most valuable. The following experiment chapters provide greater knowledge and understanding in this regard.

The present work seeks to provide solutions aimed at providing software novice developers and educational establishments with support. Accordingly, in this research, the individual elements examined are as follows:

1. Software novice developers' knowledge levels concerning usability evaluation methods (UEMs).
2. The factors potentially hindering novice developers from carrying out usability evaluations.
3. The perceived solutions with the capacity to promote evaluation methods to be integrated in the development process.
4. The effect of the devised integration solutions from the perspective of design behaviour and user satisfaction.
5. The background of the methods, models and solutions that might be suitable in assisting software novice developers' computer originations or education institutions to provide directions for future study.

6.3 Research Philosophy

The research philosophy is recognised by Sunders *et al*. (2009) as a method of knowledge development, with emphasis placed on its particular area. The researcher should garner

insight into the meaning of the research philosophy at the onset of the study; this is important for a number of reasons. As has been mentioned in the work of Easterby-Smith, Thorpe & Lowe (2002), research philosophy is pivotal in assisting the researcher in establishing the research design in regards the evidence required, and how this should be collected and analysed so as to provide answers to the study questions. Moreover, it is also essential to define the most appropriate design, whilst also recognising any potential drawbacks. Furthermore, there has also been mention to the research philosophy, which is valuable in helping a researcher to establish and even develop designs outside the scope of the individual's experience. This also could suggest how the research design is adjusted in terms of the limitations of different knowledge structures.

Regardless of whether a research is qualitative or quantitative, there is a need for its nature to be centred on some underlying beliefs in terms of what is defined as a valid research and which are the most suitable methods (M. Myers, 1997). Importantly, there are four key types of research philosophy, namely positivism, realism, interpretivism and pragmatism; Saunders *et al*. (2009) describes these four types in terms of ontology, epistemology, axiology and data collection techniques.

The present work details a number of research questions; these can be answered through the provision of positivist and interpretivist methods. Accordingly, this work implements the pragmatism philosophy with the application of a mixed-methods approach, comprising both quantitative and qualitative research. Importantly, pragmatism emphasises the research question as being the most pivotal element of establishing the research philosophy as pragmatism has the provision to work within both interpretivist and positivist (Saunders *et al*., 2009). In this same vein, it is noted by Tashakkori & Teddlie, (2008) that both practical and applied research philosophy may be presented through the adoption of the pragmatist approach, as well as mixed methods; this choice can be explained through the paradigm of pragmatism (Tashakkori & Teddlie, 2008). As is recognised, quantitative approaches are recognised simply as deduction; in contrast, qualitative approaches are induction. With this taken into account, the adoption of a mixed-methods qualitative and quantitative study means that research is able to utilise both deduction and induction, with pragmatism supporting such a route (Howe, 1988; Maxcy, 2003).

In the current work, there are three key topics that require attention, namely epistemology, positivist (quantitative) and interpretivist (qualitative).

Epistemology

The term 'epistemology' may be described as relating to the constitution of valid knowledge garnered through examination into a particular phenomenon (Cornford & Smithson, 2006). Otherwise stated, epistemology is centred on the analysis of knowledge in actual settings, and is focused on developing new theoretical models that are extensions of existing ones when considering that knowledge, and the identification of such knowledge, is not static but rather continuously changes (Grix, 2002).

Qualitative Approaches (Interpretivist)

In consideration to criticism facing the positivism philosophy, the interpretivist approach was designed (Collis & Hussey, 2013). The interpretivist philosophy is built on the foundational belief that a strategy is required in order to differentiate between social science-based objects and people; accordingly, there is a need for the researcher to ensure insight into social action's subjective meaning (Bryman, 2008). Furthermore, amongst interpretivist researchers, access to reality is only through social constructions, i.e. consciousness, language and shared meaning, owing to the fact that interpretive works seek to examine the subject under study by considering the meanings assigned to them by people (Alkraiji, 2012; M. Myers, 1997)

Quantitative Approaches (Positivist)

As highlighted by the work of Collis & Hussey (2013), positivist approaches are centred on the view that reality is independent of us and the objective is to establish theories in line with empirical research. Importantly, an objective approach is applied by positivism, which is done in order to test theories and further establish cause and effect, along with scientific laws (Walliman, 2006). Accordingly, in this paradigm, the reality is objective and independent of, or otherwise external to, the researcher; this means an objective measurement can take place through the application of quantitative research data (Collis & Hussey, 2013; Creswell, 2013). Accordingly, positivist (quantitative) research, at its core, is concerned with numbers (Remenyi & Money, 2012).

6.4 Research Strategy

Through the adoption of a mixed-methods approach, various perspectives can be integrated and thus examined with the use of at least two research strategies, meaning various elements of the investigation may be brought together (Bryman, 2007). This can be achieved through

utilising one type of research strategy or data collect approach in order to assist research utilising another research strategy or data collection approach within a single study, or otherwise applying two or more independent sources of data or data collection methods in order to provide the study's overall research findings with validation. Importantly, qualitative data is valuable in explaining the relationships between quantitative variables, whilst independent data sources are useful in contextualising the main study or otherwise to provide insight into the relative importance (Bryman, 2006).

In an effort to satisfy the study aims, a case study strategy was utilised; this is recognised as having the ability to involve both qualitative and quantitative; in this vein, it is emphasised by (Saunders *et al*. (2009) that a survey strategy can be incorporated within a case study.

In this study, the usability evaluation method has been applied in mind of two key objectives: firstly, to evaluate the framework suggested; and secondly, to garner user satisfaction. A number of factors can be impacted through usability evaluation, as noted by Bias & Mayhew (2005), such as user satisfaction and product quality.

In order to satisfy the aims and objectives outlined in this study, and to ensure the research questions are answered, whilst also ensuring an in-depth case study is provided, a combination of quantitative and qualitative methods and strategies have been chosen, including focus group, interviews, questionnaires and user testing. More specifically, through analysing the focus group, interview and questionnaire findings, triangulation was able to validate the process overall, thus ensuring strength in the conclusions; this was possible through the use of different data sources (Yin, 1994a). The triangulation strategy utilises two or more methods to confirm, cross-validate and corroborate findings within a study, and has the capacity to overcome the drawbacks inherent in one method through focus on the strengths of another (Creswell, 2003; Saunders *et al*., 2009).

The researcher is more likely to achieve a more valuable and accurate validity assessment of the research and to generalise the explanations when combing methods. Applying a mixed-methods approach can result in the adoption of both exploratory and explanatory approaches; this means gathering quantitative data following qualitative data, which is referred to as explanatory. Moreover, the exploring method is also used for the study experiments; this is valuable when seeking to design and test a new tool (Creswell, 2003).

In the present work, all of the qualitative analysis methods are centred in thematic analysis approaches, as highlighted by (Holloway & Todres, 2003). Accordingly, in the view of Braun & Clarke (2006), one of the key advantages associated with thematic analysis, in comparison

to other types of qualitative analysis, it its flexibility (Braun & Clarke, 2006). In contrast, all quantitative analyses were statistical tests involving ANOVA test, Chi-square test, frequencies and percentages.

6.5 Research Design

As noted in the work of Themistocleous (2002), the research design is acknowledged as the first element of the empirical research strategy; this involves the researcher completing an analysis of the literature so as to establish the needs and issues inherent in the research, with the most suitable research methodology then identified and used in mind of completing the study's line of enquiry in a realistic context (Thermistocleous, 2002). Yin (1994) explains the concept of research design as follows: 'an action plan for getting from here to there, where here may be defined as the initial set of questions to be answered, and there as some set of conclusions (answers) about these questions' (Yin, 1994b).

It is noteworthy to consider that the case study also is listed as a research design within the positivist approach (Alhalalat, 2005). The case study approach has been selected as the research design so as to examine the issue outlined in the research owing to the fact it is the most suitable plan for addressing the research problem. Other research designs and their overall appropriateness was considered, contrasted and assessed prior to choosing the most appropriate one. This section therefore considers and highlights the design of the case study before providing a rationalisation for this choice.

6.5.1 Case Study Design

Yin (2003, p. 13) refers to case studies as 'empirical enquiries investigating a contemporary phenomenon within its real life context, when the boundaries between phenomenon and context are not clearly evident, and in which multiple sources of evidence are used' (Yin, 2003). Moreover, the study of Collis & Hussey (2013) emphasised that, in a case study, it is recognised as most valuable to combine data collection methods so that the evidence is both quantitative and qualitative (Collis & Hussey, 2013). As such, a case study is geared towards achieving all-encompassing insight, and thus provides understanding of how the observation of the context of a specific phenomenon may be systematised (Yin, 2003).

In the present work, a multi case study is carried out in order to satisfy the study goals. In this vein, multiple-case studies are identified by Yin (1994); this involves examination into more

than one case at any one time. Moreover, Eldredge (2004) provides further support for the use of this method, particularly in relation to information systems (Eldredge, 2004). Walliman (2006, p. 46) adopts the belief that 'both quantitative and qualitative methods are appropriate for case study designs, and multiple methods of data collection are often applied (Walliman, 2006).

In this work, a number of reasons provide justifications for the selection of a multi-case study approach, as outlined as follows:

- A case study approach could be useful in defining how evaluation methods can be understood and applied by novice developers (Moore, 2006; Patton, 2002). For example, this study identified time and cost as being two complicated factors facing novice developers in their completion of evaluation methods.
- A number of different methods are employed by the multi-case approach in mind of gathering the necessary data (Patton, 2002). Such methods may be quantitative or qualitative, or a combination thereof, as in the case of the present work. The adoption of a mixed-methods approach provides a number of benefits, including the propensity to garner different types of data from various sources and analyse them in line with the research questions, and also the overall reliability and flexibility of the approach.

Despite the advantages of the chosen approach, however, there also are a number of drawbacks (Robson 1993). In actual fact, the case study approach is criticised on the basis of 'the representativeness of the findings, and whether they provide an adequate base for both the development and answering of research questions', not forgetting the researcher's influence on events (bias) (Robson, 1993). Nonetheless, in this instance, the researcher is an external investigator, meaning bias is low. Conversely, however, the researcher's opportunities to gather all the support needed, particularly throughout the process of interview, could be reduced owing to his position as an external investigator; this could also affect questionnaire distribution and access to documents. Nonetheless, the application of both qualitative methods, alongside quantitative methods and a survey-based questionnaire, may be useful in reducing or altogether eradicating bias (Alfrih, 2010).

6.6 Data Collection Methods and Analysis

In this work, both quantitative and qualitative data have been gathered through the use of a survey-based questionnaire, alongside the application of interviews, user testing and focus groups.

6.6.1 Focus Groups

The focus group approach is used primarily in mind of the design process, prototype, or group requirements amongst developers and/or users. Such sessions normally comprise 3–10 participants, all of whom sit and partake in discussions (Preece, Rogers & Sharp, 2002). It is recognised as a helpful approach and is able to encourage subjects to discuss their views on the session subject. Accordingly, such a method provides a wealth of data; nonetheless, expert moderators need to run the session in an effective and correct way (Galitz, 2007). Moreover, the focus group method is recognised as one of the methods requiring less time and effort, whilst gathering high volumes of feedback (Preece *et al*., 2002).

A focus group methodology has been gathered in order to garner insight into the requirements of users. The rationales behind its use are various. As an example, it developers insight into participants' thoughts and is a valuable way of learning; it also is regarded as an approach able to facilitate participant discussions within the group; this method further enables a brainstorm-type session considering the planned and unplanned point supporting the goals of the study (Kitzinger, 1995). Through the adoption of the focus group method, all subjects provide their opinions and ideas pertaining to the target system (Humayoun, 2011).

6.6.2 Questionnaire

When seeking to gather qualitative data, questionnaires are believed to be valuable, and commonly are seen to provide an appropriated method more so than personal interview (Bowman, Gabbard & Hix, 2002). When considering that the current research is investigative in nature, with a questionnaire a suitable instrument for gathering study data (Ardito, Buono & Caivano, 2014), it is deemed to be a suitable tool for gathering quantitative data in an effort to compile statistics (Holzinger, 2005).

In this instance, the data collection was completed through online questionnaire, which is held to be a valuable method in data collection amongst a large sample of people; however, throughout this study, various disadvantages were identified. For example, some of the subjects chose not to answer all questions, meaning they were removed from the sample.

Secondly, there is no sure way of determining whether or not the answers given were true and accurate. Thirdly, the process is both time-consuming and costly.

In the present study, two different types of questionnaire were carried out in mind of different objectives: the first centred on investigation, and posed 27 different questions. Of these questions, 13 were multiple-choice, whilst 9 implemented a Likert-type scale providing fiver different options (Saul, 2008), whereas the other questions were open-ended, and geared towards collecting qualitative data. Such questions were assigned to different sections: the first gathered demographic details; the second centred on experience with the aim of evaluating subjects' overall understanding regarding the 'design evaluation' concept and their experience of such methods, with the questions establishing whether or not subjects had evaluation-based knowledge; the third section examined the capacity of the subjects to learn and adopt evaluation methods in a professional way, with such questions assessing subjects' responses through the adoption of a 5-point Liker scale spanning strongly agree, agree, neutral to disagree and strongly disagree.

The second questionnaire focused on gathering data relating to the users' overall satisfaction of the SUS (System Usability Scale). The SUS was designed in 1986 by John Brooke (Brooke, 1996), and details 10 individual statements concerned with assessing user satisfaction and usability pertaining to software. Since its introduction, the SUS has been widely used as a reliable and short measurement. The SUS questionnaire, in this particular study, was broken down into two parts: the first comprised four demographic question; the second detailed 10 statements offering a 5-point Likert scale, providing both positive and negative statement. The score of each statement was assigned according to the Likert-scale points, i.e. –1 for the odd-numbered statement scores and –5 for the statement score for even-numbered statements. The sum of these statement scores should multiplied by 2.5 in order to get the SUS score (Lewis & Sauro, 2009; Sauro, 2011b; Usability.gov, 2013b). Importantly, aA number of works have been carried out with the use of the SUS (Bangor, Kortum & Miller, 2008; Sauro, 2011a, 2011b). (See Appendix 1: D.1 for SUS survey.)

6.6.3 Semi-structured Interview

Semi-structured interviews are able to divert away from a solid plan, meaning the interviewees are able to pose questions in order to gain interviewer clarification or otherwise to indicate new topics, thus facilitating two-way communication (Creswell, 2013). Semi-structured interviews are useful when the researcher has some insight into the interview

subjects and accordingly seeks to cover certain questions, with the number of questions potentially increasing throughout the course of the interview (Patrick, 1998). The interview method is a valuable method for garnering a wealth of qualitative data; conversely, however, it is costly and time-consuming to perform (Shneiderman, Plaisant, Cohen & Jacobs, 2013).

6.6.4 User Testing

System end users, through the application of user testing methods, are asked to carry out different tasks on the system (Humayoun, 2011). User testing is recognised as a valuable approach to establishing particular usability issues associated with design and navigation (Hasan, 2009). Furthermore, user testing is a valuable approach to establishing users' capacity to determine usability issues and thus to design solutions to such issues (Nielsen & Mack, 1994). Throughout the course of such user testing sessions, various methods may be implemented. For example, questionnaire are acknowledged as helpful when there is the aim of garnering understanding into the feelings of users throughout testing, and thus measuring overall product satisfaction (Bargas-Avila, Lötscher, Orsini & Opwis, 2009). Further, various authors in the field have made reference to the view that interviews and questionnaires are valuable in gathering data relating to user satisfaction (Nielsen, 1993b; Preece et al., 2002).
In this work, both quantitative and qualitative data were gathered through the testing sessions. Thus, the user satisfaction questionnaire was used in order to gather quantitative data. Furthermore, qualitative data were gathered through user testing. This latter approach is deemed suitable in the gathering of problems, and to comment on and make recommendations concerning participants (Usability.gov, 2013a). Accordingly, some solutions were suggested and comments made by study participants following the testing session. In order to analyse both types of data, statistical testing and thematic approaches were implemented.

6.6.5 Data Analysis

In this research, in order to analyse the quantitative data, statistical tests, including frequencies, average and number of testing such (as chi-square), were used. Moreover, the SPSS analysis software tool was applied to examine the quantitative data. In line with the first investigation, which was a questionnaire, the analysis sought to establish whether or not there was a statistically significant difference between novices' and expert software

developer ratings concerning the overall perception of understanding and practice usability evaluation. Moreover, it also seeks to determine whether there is a positive link between the knowledge of a developer and the application of UEMs in relation to technical knowledge. Further information was available to provide further understanding when considered necessary (see Appendix 1: A1 & A2).

Thematic analysis may be considered as a foundational method in the field of qualitative analysis, and may be explained as an approach utilised in mind of establishing, analysing, extracting and reporting themes and accordingly organising and describing such themes (Braun & Clarke, 2006). In the view of Braun & Clarke (2006), one of the key advantages associated with the use of thematic analysis, when contrasted alongside other types of qualitative analysis, is flexibility. Further, thematic analysis is useful in establishing new themes in the data, and provides validation concerning existing themes, including knowledge components and determinants (King & Horrocks, 2010). The figure below provides an example of thematic coding in relation to the present study's data.

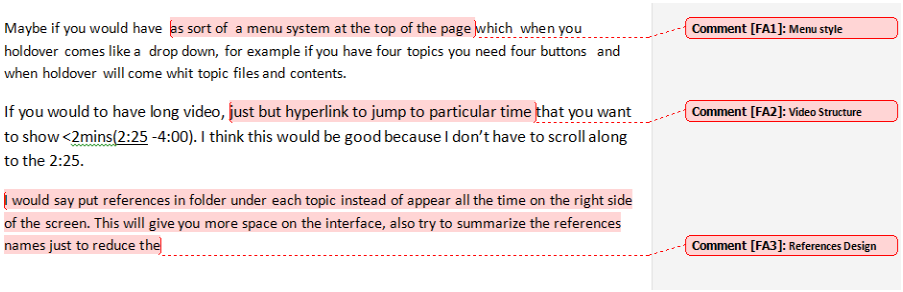


Figure 6-1: The process of establishing and detailing themes and codes

6.6.6 Validity and Reliability

In the view of Yin (2003), construct validity is recognised as a means to establishing appropriate operational measures for the subject under analysis, and further suggests different methods to allow the researcher to improve levels of construct validity when completing case studies. The research's overall validity is essential. In the present work, there was the use of a multi-case study, with this methodology applied on the basis of it being a valuable approach to ensuring research validity (Yin, 2013). The second approach considered in the work of Yin (2003) is the use of multiple sources of evidence; this should be performed so as to encourage convergent lines of inquiry—referred to as triangulation. Through triangulation, any bias in the research findings—which is common in qualitative research—can be overcome (Ryan &

Bernard, 2000). The meaning of triangulation is multi-methods; therefore, data collection methods were applied in order to gather and the necessary data.

In this work, a questionnaire was performed in mind of gathering insight into user requirements, as considered in the investigation chapter, as well as for user satisfaction. Validity in relation to questionnaire refers to the overall ability of the tool to measure what was intended (Saunders *et al*., 2009). Questionnaire design is an essential element in ensuring validity. Accordingly, it is imperative that questions are well worded, making use of terms that are likely to be familiar to respondents; this helps to improve questionnaire validity.

In regards reliability, this may be explained as ensuring the same conclusions or findings are established by any researcher carrying out the same research (Yin, 2003). Construct reliability is focused on minimising, to the greatest possible extent, a study's bias or error. The application of a mixed-methods approach provides a number of benefits, including reliability, flexibility, and the opportunity to garner different types of data in order to investigate the present situation from various organisations and resources in mind of providing answers to the research questions (Alfrih, 2010). Furthermore, in an effort to ensure bias was circumvented, the research sample population was chosen randomly.

In regards questionnaires, reliability, in this context, is described by Field (2009) as being 'a measure should consistency reflect the construct that it is measuring'. In terms of reliability, the most common approach is Cronbach's Alpha, where an acceptable value is deemed to be 0.7–0.8; an unreliable measure is anything falling much lower (Field, 2009). SUS has been found to be more reliable and has the ability to identify differences amongst smaller sample sizes than when there is the application of home-grown questionnaires and other commercially available ones (Sauro, 2011b).

6.7 Research Samples

Gathering quantitative data is the key underpinning to social research (Moore, 2006); therefore, choosing a sampling for study is critical. Sampling is focused on choosing a group representative of the population under investigation (Robson, 1993). Sampling is needed in order to achieve results with minimal effort, cost and time, although the sample cannot be too small or the data volume and detail will compromise the study objectives.

Sampling techniques can be broken down into two different types, namely probability or representative sampling, and non-probability or judgemental sampling; the sampling approach chosen depends on the research questions posed. Non-probability or non-random

sampling provides a number of different techniques, which facilitate investigators in choosing their samples in line with subjective judgement (Saunders *et al*., 2009). Quantitative sampling is more likely to be random; qualitative sampling aims at choosing a particular sample of participants in the gathering of more detailed data so as to ensure the research questions can be answered (Miles & Huberman, 1994).

Convenience sampling is one type of purposeful sampling in which the selection of a sample is based on time, money, location and/or the availability of case studies. Such a method is referred to as convenience sampling owing to the involvement of participants being based on their availability, with consideration afforded to their background.

In the present work, the sampling is non-probability. The online questionnaire was implemented in mind of examining the knowledge of software developers in relation to usability methods and their actual application of usability methods, with the objective to determine awareness resources that could promote software engineering in the completion of usability evaluations. Convenience sampling was used, comprising 84 participants in total (as can be seen in the following Table 6-1); all of the individuals had a software engineering background and various levels of programming experience. The main criteria for the sample selection were as follows: subjects needed to have a background in computing, as well as any level of programing experience. Moreover, qualitative and quantitative data needed to be collected. In the case of the former, there is a need to ensure data saturation, which can be achieved by adding additional interviewees until saturation is achieved (Saunders *et al*., 2009).

Table 6-1: Overview of methods, sample size and type of analysis

Method	Number	Qualitative	Quantitative
Investigation questionnaire	84	✓	✓
Focus groups	13	✓	
interviews	30		✓
User testing	69	✓	

6.8 Summary

The choice of research methodology indicates how research should be carried out in mind of establishing new knowledge and understanding (Saunders *et al*., 2009). The present work

implements a pragmatism philosophy, utilising mixed-methods, in an effort to provide answers to the research questions and to accordingly satisfy the research objectives.

Combining quantitative and qualitative methodologies, a mixed-methodology was adopted. Triangulation was carried out through the adoption of four different methods of data collection, namely focus groups, questionnaires, semi-structured interviews and user testing. For this study, exploratory and exploratory mixed-methods strategy was adopted.

This study's quantitative phase was made up of quantitative data collection and data analysis, with the qualitative phase of the work made up of qualitative data collection through the application of semi-structured interviews, and subsequent data analysis (See Figure 6-2).

This chapter has provided a discussion on the various data collection and analysis methods applied in the present work. A number of methods were carried out, including focus groups, questionnaire survey, semi-structured interviews and user testing, in a bid to garner empirical data. The following four chapters will be concerned with providing full explanations of such data.

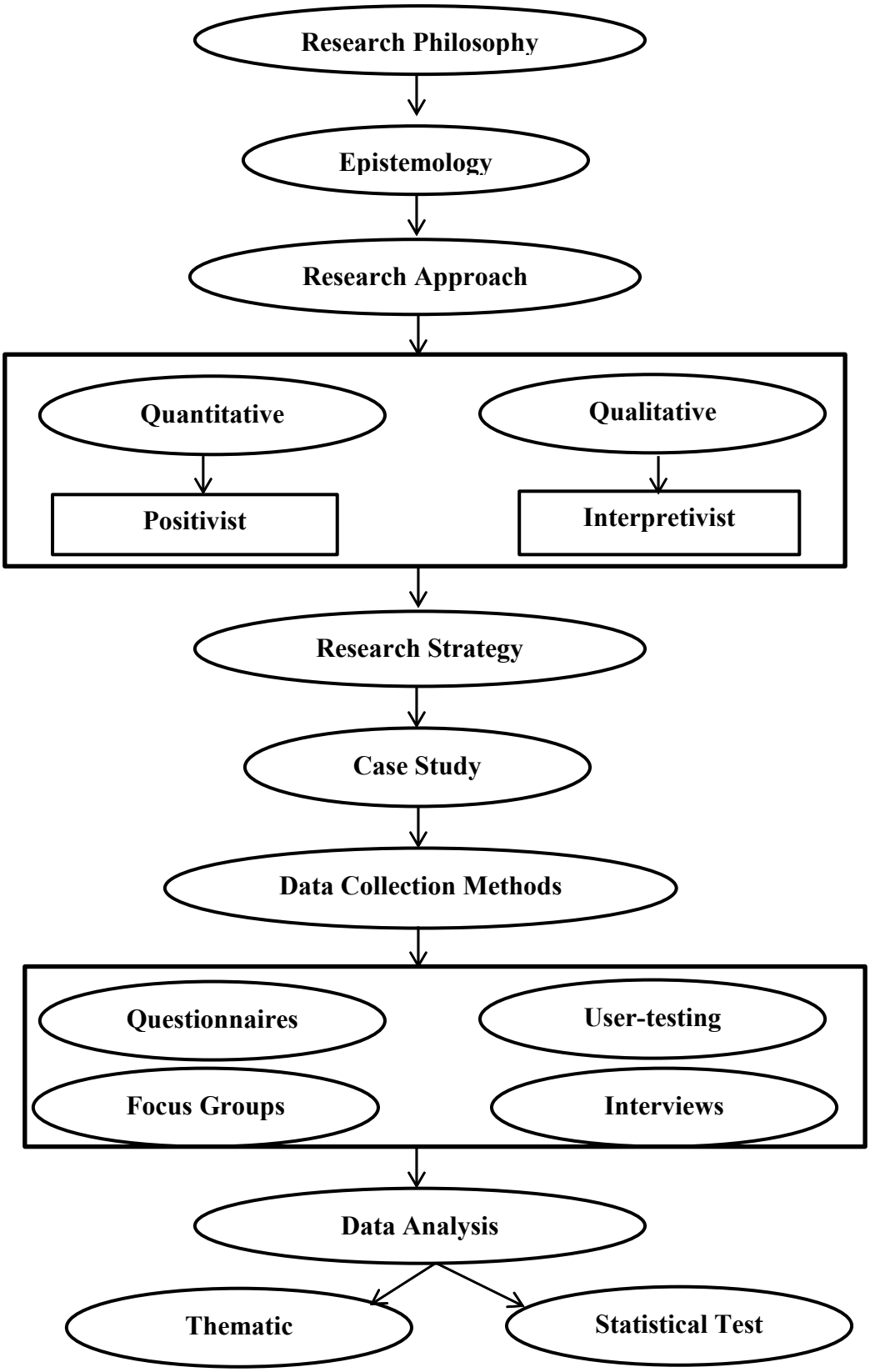


Figure 6-2: Research Methodology framework

Chapter Seven: Novice developers Investigation and Specification of New Tool Production

This chapter provides the first investigation study of the developer's evaluation knowledge, conducted to measure the amount of knowledge software engineers have. The results of the investigation study led to meeting developers and collecting their requirements for new evaluation learning resources. Finally, this chapter has been summed up with an evaluation study to measure learning resources before their application.

7 Novice developers Investigation and Specification of New Tool Production

Chapter Motivations

The usability challenges identified by a number of authors and explored in the literature review are the first step that motivates the research to investigate the usability issues from the developer's perspective. User involvement, expensive evaluation methods, lack of learning resource and developer preferences are the main challenges that this chapter attempts to investigate and solve (C Ardito et al., 2014; Nielsen, 1995; Rosenbaum et al., 2000). This chapter provides three phases of work and each phase is created based on the previous phases findings. The main aims of this work are (1) to build on the work of previous studies in improving software usability; (2) investigating the novice developers resistance to using evaluation methods and involving users during the development process; (3) devise a solution that will encourage software novice developers to create a usable product. There are different methodologies employed to achieve the goals of each phase. Table 7-1 shows an overview of phases adopted, methodology applied, objectives targeted and outcomes achieved.

Table 7-1: chapter study's methodology, objectives and outcomes of each phase

Phase Number	Methodology	Objectives	Outcomes
Phase 1	Survey based Questionnaire	▪ To investigate the gap between the people who are or willing to be software designers and the applying of usability evaluation. ▪ To investigate the software engineer's knowledge about the practice of usability design evaluation. ▪ To investigate the software engineer's willing of conducting the usability evaluation. ▪ To investigate the relation between the software engineer's level of experience and the expertise of conducting a usability evaluation. ▪ To establish awareness resource that promote software engineering to conduct the usability evaluation.	• Learning resource is required for software engineers
Phase 2	Focus groups	▪ To collect the new learning resource users requirement ▪ To establish the first design of the learning resource	• List of learning resource requirements • First version of the learning resource

Phase 3	Usability testing	▪ To assess the (dEv) learning resource ▪ To increase usability issues ▪ To complete measurement of users' overall capacity to enhance the model ▪ To discuss user finding issues	• Learning resource usability issues • User suggestion to improve the design. • An overall of user satisfaction with learning resource
	Thinking aloud	▪ To understand users thoughts throughout the testing process	
	Observations	▪ To observe the interactions of users throughout the testing process ▪ To detail additional usability issues	
	Questionnaire	▪ To establish the degree of user satisfaction with the learning resource ▪ To collect user recommendations and devise solutions	

7.1 Phase1: Novice developers Investigation Study (Questionnaire)

7.1.1 Study Motivation

It is important to consider the literature review and empirical study findings in an effort to improve the software usability field. Based on the literature review, software usability is an important area for improvement across a number of aspects, such as theoretical and empirical, for example.

Many expert studies have been performed to identify software usability issues. These have developed and compared many different methods of achieving usability in software. Further studies have examined software development companies and their application of usability evaluation in their product production process. However, few studies have investigated the typical usability knowledge of the inexperienced developer and their ability to perform usability evaluation.

The lack of studies that investigated the developer's evaluation knowledge and ability to conduct the user interaction design evaluation is leading us to explorer this issue. Furthermore, the huge number of usability evaluation methods that already existing on the literature review also encourages us to determine which the common methods that preferred for the novice developers (see chapter 3 for more details about the evaluation methods). Additionally, it's important to know if the software designing experience impacts the level of conducting user interaction design evaluation or not.

This lack of studies is addressed by this study which explores the issue of how a inexperienced developer selects and applies usability evaluation methods in their work. The level of developer experience in the study will be tracked and the choices they make from the plethora of available usability evaluation methods will be accessed.

This study provides our work achieve the first and the fifth aim of the research (see chapter 6 section 6.2)

This study is considered empirical in nature and tests the research hypothesis. The research hypothesis suggests that 'the level of programming experience of a software engineer is independent of their knowledge and practice of usability evaluation methods'. This can be rephrased as the traditional research questions,

- Does programming experience impact participants' perceptions of evaluation methods?

- Do experienced programmers conduct usability evaluation more frequently than less experienced programmers?
- Does a participants' awareness of usability evaluation increase with programming experience?

Given the importance of usability evaluation this study attempts to access whether a novice developers knowledge and application of usability evaluation methods increases in line with their technical knowledge. Ultimately a resource will be produced with the aim of insuring that a developer's knowledge and application of usability evaluation increases at the same rate as their technical knowledge.

7.1.2 Study Methodology

This research used a quantitative and qualitative methodology to collect the study data. Survey based questionnaire was the main data collection method. An online questionnaire was used to facilitate the collecting of data from a large number of people in a short period of time. Furthermore, the questionnaire is more focuses on quantitative data though qualitative data can also be collected. Questionnaires also referred as worthy method for collecting qualitative (subjective) data and often appropriated method more than personal interview (Bowman et al., 2002). Furthermore, this study is investigative research and therefore a questionnaire is an appropriated tool to collect the study data (C Ardito et al., 2014) . An online questionnaire was used for the data collection. (See methodology chapter 6 section 6.3.1) . Furthermore, questionnaire is appropriate tool to collect quantitative data to compile statistics (Holzinger, 2005). SPSS analysis software tools used to analyse quantitative data analysis. Additional information was available interactively to provide further clarification as and when required (see appendix 1: A1 & A2).

Participants

The questionnaire was viewed by more than 200 people. The sample contained in total 84 participants (67 males, 17 females), all of whom had a software engineering background and various levels of programming experience. This sample was randomly selected which is known as convenience sampling (see chapter 6 section 6.7) The main criteria for the sample selection were subjects should have background of computing and any level of programing experience. This group of participants held a variety of occupations made up as follows: 34.5% postgraduate students; 44.0% undergraduate students; 13.1% working full-time within

the education field; and 8.4% worked in other areas. The study responses were gathered from participants in different cultures and education systems. The participants were located in four different regions: a total of 42.9% were from Saudi Arabia, 40.5% from the UK, 9.5% from the USA and 7.1% from other countries. The age of participants ranged from 18 to 55 years (mean = 1.170; standard deviation = 0.406).

Procedure

The questionnaire was advertised using email and social media networks, such as IT communities and any groups of interest. Additionally, those who responded were asked to complete the questionnaire online. The data were collected and prepared for analyses using both SPSS and NVivo analysis software tools were applied in examination of qualitative and quantitative data. The study results are presented using descriptive, chi-square testing to test the research hypothesis.

7.1.3 Study Findings

The participants in the study held a variety of occupations and had varying levels of programming experience. There were 66 students, with 37 undergraduate students and 29 postgraduate students. A total of 15 participants were in full-time employment (11 of whom were working in the education field, whilst 4 worked in industry). The remaining 3 participants categorised themselves as other, meaning they were either unemployed or retired. (See Appendix 1:A.2 for all Testing and Results)

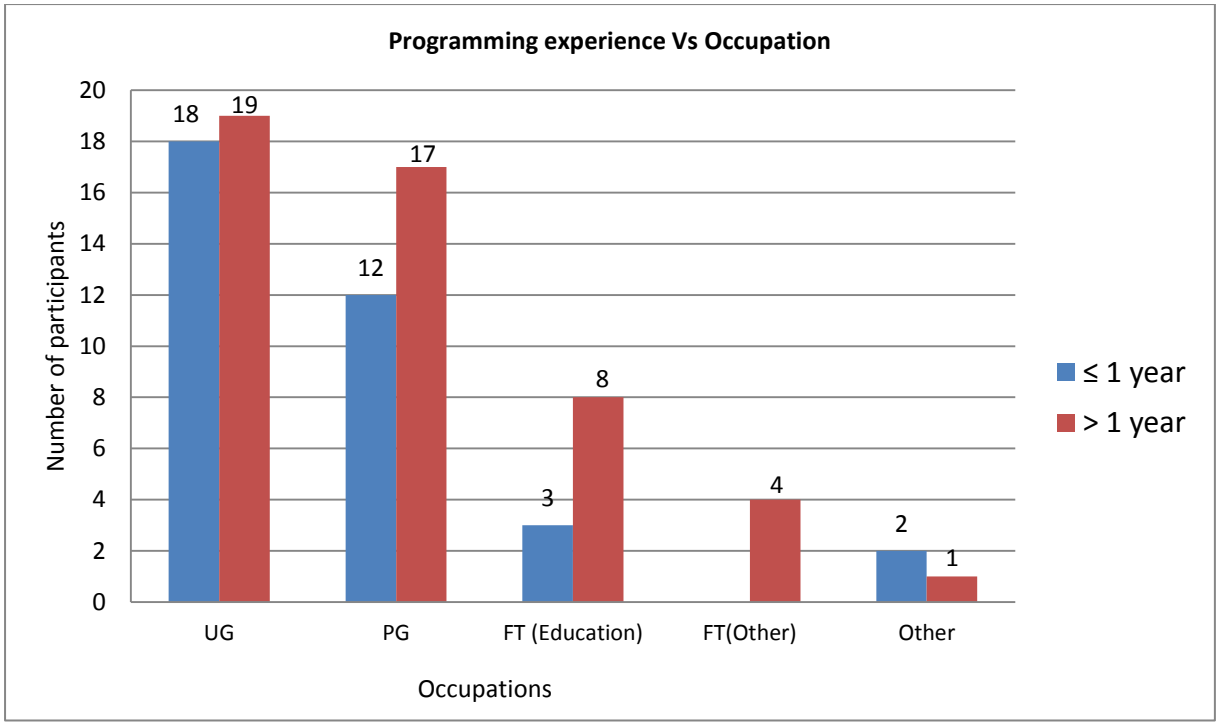


Figure 7-1: The level of programming experience of participants, categorised by occupation

The participants' experience of programming was categorised based on how long they had been programming; thus, the categories were (i) one year or less experience, and (ii) more than 1 year experience. This classification has choosing 1 year of experience as a threshold because usually in the first year of programming experience the HCI is not considered as important. However, from the second year of programming experience the HCI starts to be taken as a formal subject. This categorisation resulted in 35 participants with one year or less than one year's experience of programming, whilst 49 participants had more than one year programming experience. These two groups of participants were used to test the research hypotheses that 'the level of programming experience of a software engineer is independent of their knowledge and practice of usability evaluation methods'. The research null (H0) and alternative (H_1) hypotheses are as follows:

- (H_0): There is no significant difference in (Q_n) between the two levels of programming experience.
- (H_1): There is a significant difference in (Q_n) between the two levels of programming experience.

Q_n refers to the question number: for example, Q_1 refers to Question 1 of the questionnaire. This analysis tests the null hypothesis of the research.

Subsequently, five main results are presented. These five results use the data gathered from questions, as answered by participants. Result 1 report on the participants' perceptions of the term 'design evaluation'. Results 2–5 show a comparison of the data as it is correlated with the level of participants' programming experience. Chi-square testing was used to evaluate the impact of the level of programming experience on each question. This identifies whether or not there is a significant difference between the two groups, thereby enabling the null hypothesis to be accepted or rejected.

Result 1: Definition of the Term 'Design Evaluation'

At the beginning of the study, participants were asked to define the term 'design evaluation' in an effort to assess their overall understanding of the term. This term is loosely based on the AEA (American Evaluation Association), which defines design evaluation as the systematic approach to the gathering and subsequent analysis of data so as to define requirements, and to assess the merit, worth and significance of the item undergoing evaluation (Administrator, 2014). The question was open-ended, posed as follows: 'What does the term design evaluation mean to you?' The reason for gathering this descriptive data is based on the fact that numerous definitions of the term 'evaluation' exist. After the participants described their understanding of the term, the system presented the official definition of the design evaluation concept in order to explain the meaning of the design evaluation. It was interesting to understand how they perceived the term; however, it also was crucial to establish a clear and accurate understanding of this term as it was used throughout the remainder of the questionnaire. Based on qualitative data analysis, some of the participants did not answer the question (18 participants). Conversely, only seven of participants had a quite clear and accurate understanding of the term. Nonetheless, the majority of the respondents (39 participants) were refer to one or more of the evaluation methods as the meaning of this term for example testing and review. Moreover, 20 participants had no clear answer that not related to concept of evaluation.

In general, participants answered this question either by stating that the term meant usability testing or otherwise they responded by stating that they did not know/were not sure how to define the term. Those participants who understood the term inferred the meaning as usability testing exemplified by the use of methods, such as user testing, cognitive walkthroughs, guidelines and checklists to assess the effectiveness of the design. Other participants responded with less certainty, using terms such as 'I think', 'It may be' or 'It might be'. some

responses referred to common evaluation methods such as questionnaire and user testing however no responded referred to advance methods, such as Heuristics, for example. However, participant who gets code (PID30) was interesting and defined design evaluation as follows:

> *'Design evaluation encapsulates such a wide variety of different aspects, that it could be talked about for a very long time, so here are a few short points from me: Foremost, evaluating design entails the pursuit of discovering if it is incredibly easy for a user to figure out how the piece of software works or not. If a piece of software is created and the focus has been shifted towards what is easier for the programmer, rather than the user, there is a problem straight away in my eyes. Usability evaluations are required to test the software: to pick up particular bugs that may be present; to question the user, and possibly discover new features that could be incorporated. Were all of the requirements met? Was every piece of functionality that was planned to be a part of the final piece of software added? What could have been included if extra time was permitted? I think it is a topic that can be talked about for a very long time. To put it shortly however, for me it is about checking to see if the user experience is at a high level or not, then figuring out what can be changed to increase it if it of a low standard.'.*

Result 2: Programing Experience vs Number of Evaluations Conducted

Initially, participants were asked if they had ever conducted a software evaluation. Their answers were restricted to Yes or No values. Participants who answered No were asked to specify any reasons they had for choosing not to complete evaluations on software they had developed. The main reason preventing them from conducting the evaluation were reported as a lack of usability evaluation knowledge because it's not always required. If the participant answered Yes, on the other hand, this question then was followed-up by asking the participants to quantify the number of times they had conducted such evaluations. This question garnered multiple responses, starting from one time to four and more than four times. The data shows that 54.8% of the participants stated they had carried out the evaluation of their software on at least one occasion. Thus, 45.2% were found not to have conducted evaluations (See Figure 7-2). This result infers that more than half of the participants were sufficiently motivated to conduct the evaluation of their products. The chi-square test statistic $\chi^2(1) = 1.983$, $p > 0.05$ and the p value (0.159) were found to be greater

than the alpha level of significance of 0.05. There was no significant difference between those who conducted evaluations based on their level of programing experience.

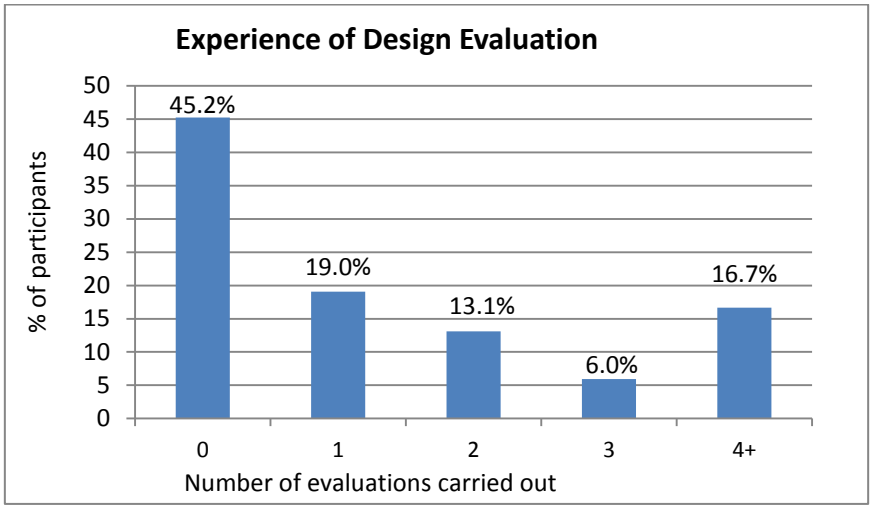


Figure 7-2: Representing how many times the participants had conducted an evaluation of their own software

Within the group of participants who conducted evaluations of their product, it was established that several different techniques were used. The most commonly applied techniques were questionnaires, task scenarios and focus groups. Future studies will garner further details about how these participants conducted their evaluation(s), how they applied the methods of design evaluation, and how they recruited their subjects. In contrast, 45.2% of the software engineers included in the study had never conducted evaluations, with the main reasons for this were reported as a lack of understanding of evaluation.

Result 3: Programing Experience vs participating in Evaluation Studies

Participants were asked if they had ever been involved in other evaluation studies similar to that of the current study. In total, 46 of the participants (54.8%) had taken part in at least one software evaluation; however, 38 of them (45.2%) had never participated. In order to establish whether or not the participants' programming experience impacted their response to this question, a cross-tabulation was produced (see Table 7-2). The probability of the chi-square test statistic χ^2 (1) = 0.269, $p > 0.05$ and the p value was found to be (0.604). This is greater than the alpha level of significance of 0.05. Therefore, the null hypothesis cannot be rejected; there is no significant difference between the answers gathered from two levels of the programming experience.

Table 7-2: An overview of the level of programming experience against the number of times participants have been involved in software evaluations, where involvement includes both conducting and participating in an evaluation

Programming Experience Level	Participation in Evaluation Study Before		Total
	Yes	No	
Less than or equal to 1 year	18 (51.4%)	17 (48.6%)	35
more than 1 years	28 (57.1%)	21 (42.9%	49
Total	46 (54.8%)	38 (45.2%)	84

Result 4: Programing Experience vs Knowledge of Evaluation Methods

At this stage of the survey based on the questionnaire, participants had already quantified their experience of conducting design evaluations. The next step in this process was to establish the techniques participants had adopted whilst conducting their evaluations. A vast number of different design evaluation techniques are available. Based on the literature (Kheterpal, 2002; Shneiderman *et al*., 2013; Vredenburg, Smith, Carey & Mao, 2002), the following list of commonly adopted techniques was used in the online questionnaire:

- Nielsen's Heuristics evaluation (Heuristics)
- The 'Thinking Aloud' protocol
- Task Scenarios
- Questionnaires
- Observation
- Interviews
- Focus Groups.

Initially, participants were asked to select which techniques they were familiar with. Subsequently, they were asked to quantify how well they knew the techniques. Their familiarity percentage (%) was quantified as being within one of the following three classifications:

- I am highly familiar with technique (60% or greater)
- I am familiar with this technique (30%–60%)
- I am not familiar with this technique (less than 30%).

Figure 7-3 shows the results. It is noteworthy to highlight that the data is ordered by participants' degree of familiarity with the evaluation method.

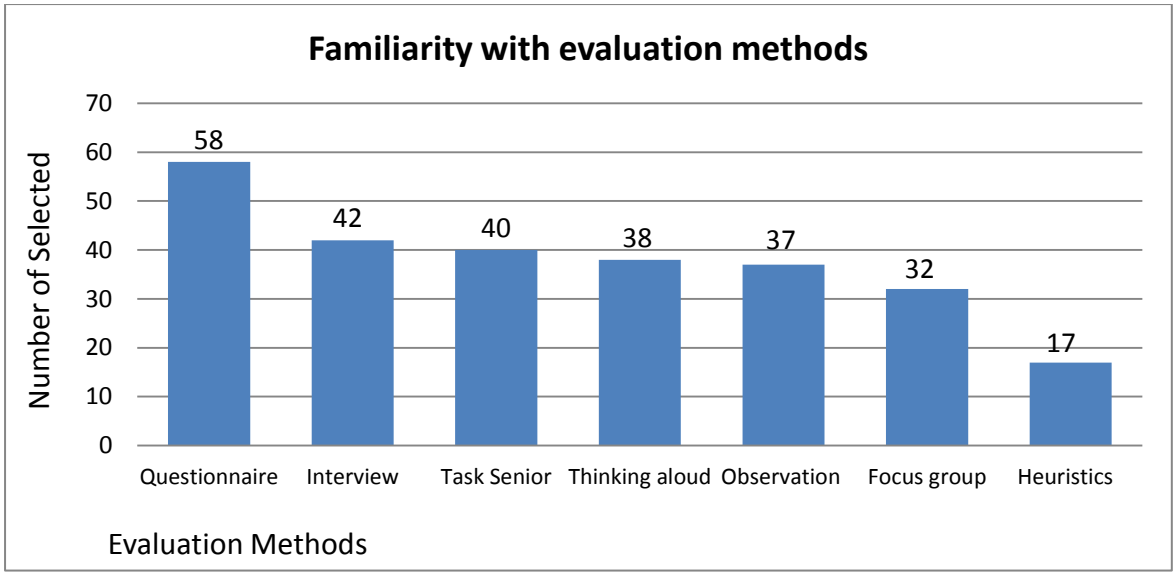


Figure 7-3: Seven of the popular evaluation methods plotted against how familiar participants were with each of the methods

Throughout the course of the analysis, the chi-square test was performed in order to establish whether there was any relationship between the programming experience of participants and the number of evaluation methods known to them. In this case, the null hypothesis testing showed no significant difference in the practice of evaluation methods between the two groups of programmers. Table 7-3 shows that the following methods—Questionnaires, Interviews, Task scenarios, Thinking aloud, Observations and Focus groups—all have p values that are greater than the alpha level of significance of 0.05. The null hypothesis for these methods cannot be rejected, showing that responses from participants were not influenced by participants' experience of programming. However, there was one anomaly: the chi-square test for Heuristics, $\chi^2(1) = 5.059$, $p < 0.05$ and p value (0.024) was less than the alpha level of significance of 0.05. Consequently, the level of programming experience is seen to influence the number of novice developers using Heuristics; therefore, the null hypothesis can be rejected owing to the fact there is a significant difference between the two levels of programing experience groups for that specific method of evaluation.

Table 7-3: The results of analysing the impact of programming experience on participants' familiarity with each of the evaluation methods. Chi-square testing was performed; the core outputs from this analysis are displayed

Method	Value	Degrees of freedom	p value
Questionnaire	0.159	1	0.690
Interview	1.224	1	0.268

Task scenario	0.087	1	0.768
Thinking aloud	0.269	1	0.604
Observation	0.399	1	0.528
Focus groups	1.131	1	0.288
Heuristics	5.059	1	0.024

Participants were asked about their understanding of the term discount usability (as established by (Nielsen, 1993b)). The majority (68 participants) were not familiar with the term (see Table 7-4). Participants asked about the discount usability term that established by (Nielsen, 1993b) and the result shows that the majority of them (68 participants) were not familiar with this term (see Table 7-4). Chi-square test was performed in order to find whether there was any impact of the programming experience on participant's familiarity with the discount usability term. The results show that experience level did not impacted the number of participants who know about this term at, $\chi 2\ (1) = 0.882$, $p > 0.05$ and the p value was found to be (0.348).

Table 7-4: Discount usability familiarity terms

Programming Experience Level	Yes	No	Total
≤ 1 year	5	30	35
>1 year	11	38	49
Total	16	68	84

Result 5: Programing Experience vs Professional Application of the Evaluation Methods

Several questions in the study explored participants' ability and perspective on the professional application of evaluation methods (See Table 7-5). This incorporated two core issues: the importance ascribed to evaluation and the availability of resources.

The Importance of Evaluation

The study explored participants' perspectives on the importance of software evaluation. Participants were asked to rate how important they considered knowledge of evaluation methods to be to software engineers. The results show that the majority (86.9%) of participants consider that it is indeed important for software engineers to know how to

evaluate their applications. A small number of participants (1.2%) disagreed that understanding evaluation methods was important. The remainder (11.9%) reported that they were undecided as to whether or not it is important to know about software evaluation. The chi-square test was carried out with the aim of assessing the influence of programming experience on the participants' rating of the importance of evaluation. The results showed that there was no significant different of participants' experience of programming on their rating of the importance of evaluation at χ^2 (2) = 1.010, p > 0.05 and p value (0.604). This result shows that both experienced and inexperienced participants had the same views about the importance of evaluation.

Furthermore, participants were asked to rate the importance of encouraging novice developers to regularly use evaluation methods. The majority of them agreed that novice developers should be regularly asked to employ evaluation methods on their product. A small number of participants (1.2%) disagreed that encouraging novice developers was required and 8.3 % of the study participants were undecided. A chi-square test was preformed to find if the programing experience impact on the participants answer. The result showed that there was no impact from programming experience level on the participants answer at $\chi 2$ (2) = 1.428, p > 0.05 and p value (0.490). Furthermore, participants were also asked if the programmers evaluation skills should be developed alongside their technical skills. The majority of participants (86.9%) agreed that evaluation skills should improve alongside technical skills, 10.7% of them were undecided and 2.4% disagreed with this statement. A Chi-square test was carried out with the aim of assessing the influence of programming experience on the participants' rating of developing the evaluation skill. The result showed that there was no significant impact of the participants' programming experience on their rating of the development of usability evaluation skills at $\chi 2$ (2) = 2.153, p > 0.05 and p value (0.341).

Resources for Learning

Participants were asked whether it would be useful to have a resource that supported their use and application of evaluation methods. The data collected shows that 89.3% supported this idea whilst 2.4% disagreed, and the remainder of the participants (8.3%) were undecided. Chi-square testing showed no significant impact of participants' level of programming experience on their rating of the importance of a learning resource at the p < 0.05 level $\chi 2$ (2) = 2.090, p > 0.05 and p value (0.352). Thus, programing experience does not affect participants' requirements for a learning resource.

This study shows that 60.7% of the participants were prepared to spend some of their time learning about the topic of software evaluation. Furthermore, participants reported that they would be satisfied using learning resources to achieve this outcome; however, 14.3% also made it clear that they were not interested in spending time on learning. The remainder—a substantial 25.0%—were undecided. The data analysis shows that the level of programming experience does not have an effect on participants' perspectives on this statement (at the $p < 0.05$ level (χ^2 (2) = 1.631, $p > 0.05$ and p value (0.442)). This test shows that programing experience did not have an influence on participants' answers to this question.

According to the data collected, 82.9% of unexperienced participants (those participants with 1 year or less experience in programming) indicated that they would take time to learn at least the basic processes of software evaluation. Additionally, 77.6% of experienced participants (more than 1 year of programming experience) also reported that they would prioritise this learning. A chi-square test was carried out. The results were at the $\chi 2$ (2) = 1.038, $p > 0.05$ and p value (0.595). Thus, there is no significant effect of programming experience on participants' overall willingness to learn basic processes of evaluation. The study shows that 75.0% of the study participants felt they would be confident in conducting an evaluation if they knew more about the process. In contrast, 6.0% did not feel confident about conducting evaluations for their applications. The remaining 19.0% felt they should be able to complete the evaluation but were not absolutely certain this was the case. A Chi-Square testing result shows that no significant different for the two programing experience groups, χ^2 (2) = 0.038, $p > 0.05$ and p value (0.981)

Participants were asked whether teaching software evaluation should be mandatory for all software engineering students. The study shows that 71.4% of participants agreed it should be a mandatory component of Software Engineering. A small number (7.1%) disagreed with this, whilst 21.4% were undecided. Chi-Square testing was carried out in order to measure the effect of programming experience on participants' responses to this question with the result for χ^2 (2) = 1.669, $p > 0.05$ and p value (0.434). This analysis showed that there was no significant different of programming experience on their responses.

Finally, participants were asked whether they would employ an external evaluator to assess their software in the future for time consuming issue. The results reported that 47.6% agreed with the statement 'I always use an external evaluator as I don't have the time to get up to speed on this topic'. In contrast, 16.7% of participants reported that they disagreed with the statement. The remaining 35.7% were unsure. A Chi-Square test was conducted to compare the effect of programming experience on the use of an external evaluator. Participants were

provided with the statement, 'I always use an external evaluator as I don't have the time to get up to speed on this topic'. This analysis showed no significant effect at $\chi^2(2) = 0.421$, $p > 0.05$ and p value (0.810). This result demonstrated that, for this sample, experienced and inexperienced participants rated the statement similarly.

Table 7-5: Questionnaire Statements

	Experience level	**Agree**	**Neutral**	**Disagree**	**Total**
Statement 1	≤ 1 year	31	4	0	35
	>1 year	45	3	1	49
	Total	76	7	1	84
Statement 2	≤ 1 year	30	5	0	35
	>1 year	43	4	2	49
	Total	73	9	2	84
Statement 3	≤ 1 year	26	7	2	35
	>1 year	37	9	3	49
	Total	63	16	5	84
Statement 4	≤ 1 year	18	2	5	35
	>1 year	22	18	9	49
	Total	40	30	14	84
Statement 5	≤ 1 year	33	2	0	25
	>1 year	42	5	2	49
	Total	75	7	2	84
Statement 6	≤ 1 year	29	5	1	35
	>1 year	38	7	4	49
	Total	67	12	5	84
Statement 7	≤ 1 year	23	9	3	35
	>1 year	28	12	9	49
	Total	51	21	12	84
Statement 8	≤ 1 year	30	5	0	35
	>1 year	43	5	1	49
	Total	73	10	1	84
Statement 9	≤ 1 year	26	8	1	35
	>1 year	34	10	5	49
	Total	60	18	6	84

Statement 1: Students of software engineering should be encouraged to use methods of design evaluation regularly.
Statement 2: Ideally, software engineers should develop their technical skills alongside their design evaluation skills.
Statement 3: Once I learn how to run design evaluations, I will be more likely to do them.
Statement 4: I always use an external evaluator as I don't have the time to get up to speed on this topic
Statement 5: Resources that teach you how to evaluate your design are important.
Statement 6: I want to learn the basics about design evaluation so I can get on and do it as soon as possible.
Statement 7: I want to invest my time in order to learn all about the topic of design evaluation.
Statement 8: It's important for every software engineer to know how to evaluate their software.
Statement 9: Teaching on design evaluation should be mandatory for all software engineering students.

7.1.4 Discussion

The main hypothesis of this study proposes that the level of experience of a software engineer does not affect their knowledge and practice of software evaluation. Thus, as they develop their expertise in software development, they do not necessarily develop their skills in software evaluation to the same extent. Thus, in general, their level of programming experience does not impact their knowledge of evaluation methods. Based on the analysis presented in this study, this hypothesis appears to hold with a single exception: the use of the Heuristics evaluation method.

The online questionnaire was used to collect data because this is an investigation study and seeks to garner numerical data with little qualitative data. Thus, a questionnaire method is the best way of gathering both quantitative and qualitative data from large participants on different regions. The study sample is not that large (84 participants); however, this study had some participants' specifications, such as computer science major work and level of programming experience. In this study, a total of 78.5% of the participants were undergraduate and postgraduate software engineers. Therefore, the data within this study comes from students currently facing the need to evaluate their software applications.

When asked to define design evaluation, responses indicated that many (20) participants were unsure about the meaning of design evaluation; some had a vague idea but, in the main,

they were unclear. Thus, it is likely that this group of participants had not encountered design evaluation before. These differences in definitions are expected to be identified because no definition has been considered as a unified definition for the design usability evaluation. Furthermore, the evaluation can be defined from different aspects and study fields. These inaccurate and highly diverse definitions of design evaluation clearly highlight the fact that greater clarity is required. Ardito et al. had conducted two studies in two different geographical contexts and they reported different results of usability evaluation understanding. One study reported that the majority of study participants showed a clear understanding of the term usability evaluation. In the other study in a different geographical context the majority of participants where unclear about the meaning of the term "usability evaluation" and substituted functionality testing for true usability testing (C Ardito et al., 2014). It can be concluded that without education in the true meaning of usability evaluation must software novice developers will confuse "functional software" with "usable software". For the field of software usability education of novice developers is therefore critically important to the production of truly "usable" software.

The study results have been produced based on participants' level of programming experience. Participants have been divided into two groups of experience: unexperienced and experienced. The unexperienced group contained all participants with a year or less than one year programing experience; however, all participants with more than one year were considered as experienced participants. This divination aimed at classifying the study participants based on their level of experience. Moreover, this classification provides 1 year as threshold to include all undergraduate years and above. Based on these two categories of programming experience, the research hypothesis was tested and the results have been produced.

This study investigated participants' familiarity and use of seven different evaluation methods. These evaluation methods have chosen based on the most widely used and inexpensive methods reviewed by authors in the evaluation field (Kheterpal, 2002; Shneiderman *et al*., 2013; Vredenburg, Smith, Carey & Mao, 2002),. The study aims at exploring how the most widely used evaluation methods are familiar to the study participants. Our study found that software novice developers were familiar with the more common evaluation methods, such as questionnaires, as expected due to their widespread use. However, there was significantly less clarity and experience with the more advanced methods (such as Nielsen's Heuristics, for example). It can be deduced (see Figure 7-3) that participants are familiar with all the methods with different level familiarity however

Heuristics evaluation is the less. This means participants have a working knowledge of evaluation methods; nonetheless, it may be the case that they still need to know how to implement these methods in detail. Overall, questionnaires were the most popular choice of evaluation method with more than two-thirds of the participants reporting its use. The simplicity and ease of using questionnaires could be the reason behind why participants are more familiar with this method. Participants reported that they were reasonably (30%–60%) familiar with the majority (70%) of evaluation methods (five out of the seven methods). However, Nielsen's Heuristics method was the least familiar amongst the participants of the study, with only 20% of the subjects indicating familiarity. More advanced evaluation methods are used less by the participants in the study; this could be ascribed to the difficulty associated with their use. However, this also could be due to a lack of clear, concise information about the method. The weak use of Heuristics shows that software engineers need to improve their knowledge about other methods. The researcher considers this a strong finding that warrants future examination.

A large proportion of the participants totalling 45.2% avoided the issues of software evaluation by never conducting or participating in any evaluation sessions. This problem of not involving the user within the development of software has also been identified commercially (Ardito, Buono & Caivano, 2014). These businesses ascertained that developers did not include participants during the requirements-gathering stage as it was perceived to be a waste of time since these interactions complicated the development. Furthermore, it was perceived that users were unclear about their needs, were imprecise on occasion, and sometimes asked for additional/changed functionality as a result of such consultations. Moreover, our study also found that many (47.6%) of the participants used third party evaluators due to time pressures rather than conducting evaluations themselves. This is quite a remarkable conclusion. However, some studies supported that inexperienced people can conduct the evaluation and identify usability problems if they are encouraged by tools or training (Bruun & Stage, 2014; Howarth et al., 2009; Skov & Stage, 2005). Based on this result, it appears that, for our group of participants, experience in software development does not imply experience in software evaluation. Note that the experienced developers in this study reported proactively avoiding the task of software evaluation; hence, they were not adopting methods that would have led to the development of more usable software. Future research needs to find out how to reduce these two percentages, i.e. 45.2% of avoiding evaluation and 47.6% of using external evaluators. Future research also needs to promote them to self-evaluation.

Self-evaluation completion is appropriated for some reasons. Firstly, self-evaluation to let novice developers develop understanding pm how the application works. Thus, conducting the evaluation also attempts to reduce novice developers' mindset by allowing them to involve users during the development process in order to stop thinking as both user and developer, which is considered a challenge for avoiding self-evaluation completion (Ardito *et al*., 2014; Bak *et al*., 2008). Secondly, the lack of evaluation knowledge is another issue preventing self-evaluation completion (Rosenbaum *et al*., 2000); however, training is a way of improving knowledge and experience concerning the evaluation and allows them to self-conduct evaluation sessions (Bruun & Stage, 2014; Howarth *et al*., 2009; Skov & Stage, 2005). Thus, self-evaluation completion will allow developers to improve their knowledge by using learning resources before they start running evaluations. Thirdly, the cost of evaluation is mentioned as a reason preventing developers from completing evaluations (Bak *et al*., 2008). Thus, developers can conduct evaluations by using low-cost evaluation methods, which provides a solution to reducing the costs of evaluation (Gutwin & Greenberg, 2000; Nielsen, 2007). Furthermore, involving users by using evaluation methods will reduce the costs of evaluation in the future (Bevan, 2005).

On the other hand, confirmation bias may be considered the greatest disadvantage of self-evaluation completion. Confirmation bias refers as 'the human tendency to search for, collect, interpret, analyse or recall information in a way that confirms one's prior beliefs or preferences', also is defined as a serious problem for decision validity (Jorgensen & Papatheocharous, 2015). Furthermore, some empirical studies mention that testers often are choosing only positive results (Calikli, Aslan & Bener, 2010). Thus, self-evaluation is an opportunity for developers expecting results in advance, and leads on their belief. However, developers should be concerned with the dengue of conformation bias, and also should understand the information against their belief (Dooley, 2013).

Evaluation topics were of interest to the majority of the participants, regardless of their level of programming experience. Clearly, they showed some interest in the topic, even by participating in this study (which received no reimbursement). This engagement in the topic should encourage the utilisation of a future learning resource. When asked about the importance of a learning resource based on evaluation, the majority (89.3%) of participants

rated it as important. Thus, this result focused the research towards the development of a learning resource aimed at supporting software engineers in their practice of software evaluation.

Moreover, both experienced and unexperienced are willing to learn how to evaluate and conduct the evaluation for their products by themselves. This results show 77.6% of experienced participants and 82.9% of unexperienced participants are interested to learn even the basic elements of design evaluation. These results give evidence to support that software engineers are able to accept the discount usability concept; however, the lack of learning resources providing inexpensive and quick methods prevent them from conducting an evaluation. Learning resources that teach novice developers how to evaluate their products will be a great solution centred on managing the misunderstanding of evaluation methods and techniques: 60.7% of the study participants were interested in spending time looking at this learning resource and learning from it, as created and based on their requirements and needs. Thus, learning resource creation should be carefully designed so as to promote novice developers in its use; otherwise, there is no benefit to be garnered from it. Furthermore, 75.0% of the study participants are willing to run the evaluation by themselves if they have enough knowledge about how to conduct the evaluation in the right way. In the study, we asked participants if they agree to have evaluation methods as a compulsory module for all undergraduate novice developers. A high percentage (71.4%) agreed with this suggestion. This means participants can see the benefit of the evaluation topic on their experience, which is a great finding point before starting to plan for any learning resource.

Learning resources are required to encourage software engineers using evaluation methods during their development process. A total of 89.3% of the study participants agreed that learning resources should be built to them; however, this resource should be created based on developers' needs and structure, otherwise they would not get any benefits from them. According to Skov and Stage (2012) undergraduate students proved able to conduct usability evaluation and complete usability reports after they had engaged with an appropriate training course (Skov & Stage, 2012). The researcher expects that learning resources would impact novice developers' thinking more in regard to their design usability level as novice developers who self-conduct evaluation sessions will come up with tangible results that improve the design, whilst also avoiding previous mistakes on future development.

In general, this study shows that novice developers' experience of programing does not impact the level of evaluation knowledge, which means all developers with different levels of

experience have similar levels of knowledge and experience relating to evaluation methods. Therefore, one improvement strategy could affect any developer who is interested in being a self-evaluator. However, this strategy should be considered from different viewpoints in order to ensure greater effectiveness on the usability field. Firstly, learning resources should be considered as the final product with which novice developers interact. Secondly, novice developers with different levels of programming experience should be the main source for new learning resource requirements, including designing and evaluating the learning resource. Thirdly, both experienced and unexperienced people should be able to use the learning resource; however, it would be preferable to focus on the new developers (undergraduate students) to start building their evaluation knowledge at the same time as programing knowledge whilst avoiding the mind set challenge. Finally, evaluation knowledge improvement should be the main goal of learning resources, which should be seen as a way of changing the common meaning of evaluation, which is 'testing' at the end of the project to ensure wider meaning than mere testing. Thus, these four suggestions should be considered in mind of improving the level of evaluation knowledge and accordingly promoting the use of evaluation methods.

7.1.5 Conclusions and Outcomes

Overall, this study has shown that software engineers are willing to have more support in evaluation; however, it also shows that software engineers require support in order to extend the practice of software evaluation. The vast majority of the participants (90.5%) agreed with the statement 'Students of software engineering should be encouraged to use methods of design evaluation regularly'; however, almost half (45%) of the group had never performed an evaluation of their own software. The study has shown that time is a major constraint. Participants reported that their primary reason for not conducting design evaluations was due to a lack of time. Despite this, however, many participants (60.7%) stated that they would be convinced to spend time learning more about how to conduct evaluations provided that a specifically targeted learning resource was available. Participants repeatedly reported that they did not want a vast resource that presented every possible method.

Overall, it is the view of the authors that the topic of software engineering requires the inclusion of a stronger emphasis on software evaluation. The next stage of this research will support this endeavour by developing a learning resource. This resource will enable software engineers to exploit the benefits of evaluation within a specific timeframe defined by

themselves. Initial work already has shown that participants are interested in such a resource provided it is clear, concise and not overly time-consuming. Therefore, the development of an appropriate resource is underway.

7.2 Phase 2: Novice developers Requirements Collection

7.2.1 Study Motivation

After the first study has been carried out, with the presentation of finding and various recommendations, this study has been created to achieve one of these recommendations. This study is referred to as Design Evaluation Users' Requirements. The aim of the study is centred on collecting user requirements about design evaluation and accordingly building a resource based on these needs. This study seeks to obtain more details about users' needs, face-to-face, in mind of building the resource.

7.2.2 Study Methodology

This study aims at gathering in-depth information and coming up with a list of requirements. Thus, qualitative data needed to be collected through the use of a qualitative data collection method, such as interview, observation or focus group, for example. This study adopted a focus group methodology to collect insight into user requirements. The reasons behind using this method include the following: it is a sound way of learning and understanding participants' thinking; it is a valuable approach to encouraging the participants to talk within the group; and this method allows a brainstorm-type session that discusses the planned and unplanned points supporting the study goals(Kitzinger, 1995). The appropriated number of focus group participants has suggested to be between 4 to 12 participants (Tang & Davis, 1995). However, smaller groups are preferable to make the session easy to conduct.

Participants

The sample contained 13 participants (11 males and 2 females), all whom had had degree of computing with a variety of different levels of programming experience. Furthermore, all participants were Plymouth university students, where they were enrolled on different computing courses. The main criteria for the sample selection were as follows: subjects should have programming experience; and the subjects should have a minimum of one designing product and be able to learn more about the user interaction design assessment approaches.

Procedure

This study provided three focus group sessions that divided participants into three groups: the first group comprised 4 participants; the second group had 4 participants; the third group included 5 participants. Each session took 1 hour of discussion. All participants were already aware of the goals behind the study and focus groups; however, the research investigator started by providing an introduction about the study aims and procedures. Participants were asked to sign the consent form at the beginning of the session, and recoding permission was taken. The session moderator asked general questions about the evaluation, usability evaluation methods and discount usability. Subsequently, participants were asked to give their requirements for the learning resource and structure. In each session, the moderator classified participants' list of requirements by using the white board to make sure all requirements were listed and clearly discussed. At the end of the session, participants had their participation card, which included the participation code used for additional information and data withdrawal. The study research analysed all three focus group sessions to come up with a list of resource requirements and associated structure.

7.2.3 Tool Design Specification

Each focus group was asked to discuss several questions with the aim of generating a list of items for inclusion in the learning resource. Each session devised individual suggestions of requirements (see Figure 7-4).

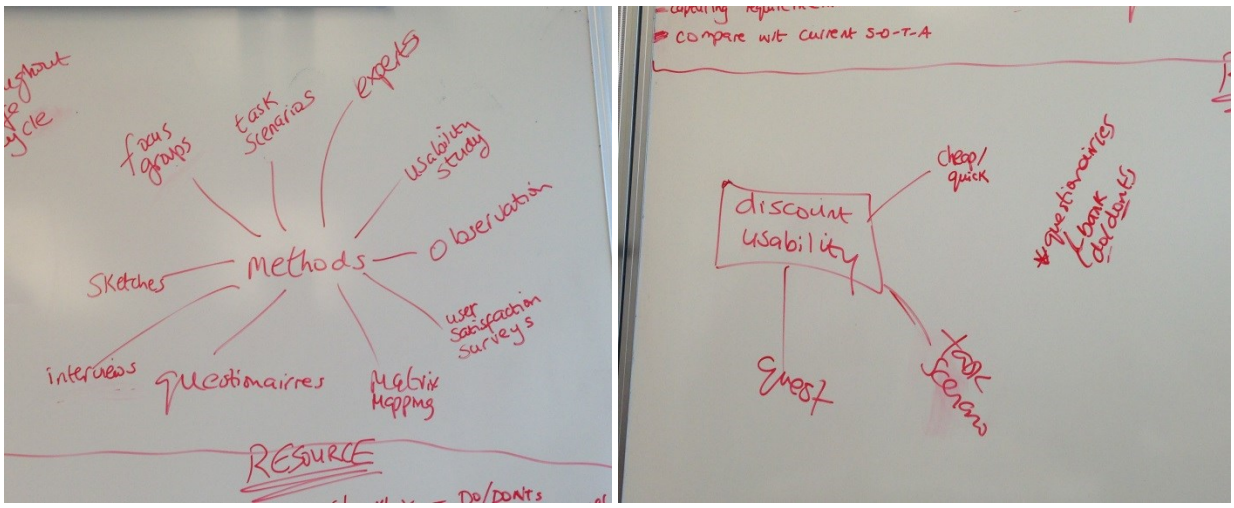


Figure 7-4: Focus group sessions

This study shows that the majority of the participants preferred an online learning resource. The focus groups filtered the suggested items for inclusion in the learning resource. The following is a list of the items required in the design evaluation learning resource.

- User groups link with other groups to establish valuable samples: these are used for the evaluation or requirement collection.
- Screen recording: methods centred on recording screen activities throughout the user testing phase.
- Standards and general guidelines.
- Usability percentage, such as through the provision of a tool able to measure usability and accordingly provide a percentage-based result.
- Checkbox list in mind of assisting the developing in checking the design.
- The provision of good and bad examples to aid design.
- Task scenario templates.
- Ethical approval.
- Video guidelines.
- In-depth data provided through an information page.
- Audio guidelines, such as through a podcast.
- Step-by-step process breakdown.
- Usability group in order to provide developers and users with assistance.

Focus groups also suggested the design evaluation methods that should be included in the learning resource (see the following Table 7-6). All participants agreed that a number of evaluation methods should be included with more details to help software engineers correctly apply these evaluation methods. Table 7-6shows a mix of automated methods such as screen recording and non-automated evaluation methods such as user testing, questionnaire and so on that where identified by the study participants.

Table 7-6: Evaluation methods

Evaluation Methods
User Testing
Recording (screen, time)
Measuring (use liker scale to rate the software and give feedback)
Questionnaire
Observation
Focus group
Cognitive walkthrough
Interview.

7.2.4 Discussion of Design Specification

This study clearly shows the new online learning resource is supported by software engineers owing to the advantages recognised from such a resources. Based on the study result, participants were provided with a list of requirements that support three main concepts: easy and quick, technology involvement and expert assistance.

Firstly, the numbers of participants' requirements were easy and quick things to learn and to do, such as 'Short videos as guidelines' and 'scenarios templates', for example. This clearly shows that software engineers are not interested in spending long periods of time learning. Furthermore, providing good and bad examples of usability is an easy way of teaching of the advantages of evaluation methods in the context of software usability. In summary, the new learning resource should be easy to understand and provide concise and important information. Nielsen is considered as one of the first authors who believed that quick and cheap evaluation methods are appropriate for usability evaluation, thus he established the "discount usability" term (Nielsen, 2007).

Secondly, participants mentioned high advanced technology should be provided by the new learning resource, such as in the form of 'Screen Recoding' and automated methods, for example, with the ability to give 'a percentage of usability' for the design. This means participants want this learning resource to help them with advanced techniques that they otherwise cannot do or that are difficult to have. In summary, this resource should be able to teach users the easy evaluation methods and serve with advance techniques.

Thirdly, participants mentioned getting in touch with experienced evaluators. This would be a valuable approach and would save their learning time and efforts by involving experts in the resource. Thus, this service suggested an online discussion group that is available to all resource users. Furthermore, a list of links containing more in-depth information should be available for users. In summary, the new learning resource should act as a reference for all users and should be valuable in fulfilling their requests.

The participant list of the evaluation method suggested for inclusion clearly shows that developers have knowledge about the common evaluation methods. This list of methods contains both quantitative and qualitative data collection methods, both of which are available as inexpensive or expensive methods. However, the new learning resource aims at only inexpensive, easy and common evaluation methods, meaning some of them will be included whilst others will not.

7.2.5 Conclusion and Outcomes

This study has been conducted to collect the evaluation method learning resource requirements. The focus group method was used to collect the study data and achieve the study goals. At the end of three focus group sessions with 13 software engineers, the study was successful in generating a list of requirements that participants would be interested in being fulfilled through a resource. Thus, the next step is to plan and develop the resources based on these requirements.

The development of the learning resource will be carried out across two stages, with the first focused on providing detailed information about the selected design evaluation techniques. The second added the requested supporting material (ie. Wikipedia links and short videos). Some functionality, such as online usability groups, was listed, which have already been fulfilled through common online software.

7.3 Phase 3: Tool Prototype Design Evaluation Study

7.3.1 Study Motivation

The learning resource is referred to as Design Evaluation Learning Resource, and known as the 'dEv' resource. Thus, the dEv term will be used when considering the resource in future works. Our dEv learning resource and framework have built to prompt the novice developers for using evaluation methods during the development process and increase the software usability (Almansour & Stuart, 2014). This study we measure the suggested learning resource to investigate the following points: (1) the learning resource usability issues. (2) Participates ability for making decision on the dEv learning resource. (3) User satisfaction the dEv learning resource.

7.3.2 Study Methodology

This work applied usability assessment approaches involving users in evaluating the design and accordingly gathering dEv-related feedback. Accordingly, there has been the use of four common approaches at various stages of the usability evaluation process, namely observations, thinking aloud, questionnaire and user-testing (Holzinger, 2005). Table 7-7 details the various testing methods applied in this work, and the various objectives underpinning their use. User testing enabled participants to carry out tasks on the real dEv product. Such a method was applied as one of the key aspects in usability methods that results in users being able to interact with the real design and deliver critical data whilst also increasing usability issues (Nielsen, 1994). Moreover, user testing is a suitable method to establishing the capacity of users to establish usability issues and accordingly propose solutions (Nielsen & Mack, 1994). Thinking aloud also has been applied in order to encourage participants to communicate what they are doing. Participants' thoughts throughout the user testing stage aids in the establishing of system errors and the root causes behind issues (Holzinger, 2005). Observation was also adopted in order to focus on the interaction of users throughout the user-testing process. This method is useful in establishing the key usability issues and accordingly creating a usable user interaction design (J. D. Gould & Lewis, 1985). The questionnaire method was implemented in mind of gathering data pertaining to user satisfaction with our dEv resource. Questionnaires are recognised as valuable when striving to gain insight into users' feelings throughout testing and accordingly measuring their degree of product satisfaction (Bargas-Avila et al., 2009). Moreover, the

questionnaire is regarded as a suitable method for gathering quantitative data to compile statistics (Holzinger, 2005). Nielsen states that, 'the first several usability studies you perform should be qualitative' (Nielsen, 2006); thus, we align with this idea and accordingly include three methods in an effort to gather qualitative data opposite to one quantitative method to establish the user satisfaction of the model. Both methods have been used to strengthen the results (R. B. Johnson & Onwuegbuzie, 2004); furthermore, a mix-methods approach would be useful to expand the scope of the study findings (Sandelowski, 2000).

Table 7-7: Research method phases for the dEv usability testing

Phase No	Phase denomination	Purpose and achievement
Phase (1)	User Testing	▪ To assess the (dEv) learning resource. ▪ To increase usability issues ▪ To complete measurement of users' overall capacity to enhance the model ▪ To discuss user finding issues
Phase (2)	Thinking aloud	▪ To understand users thoughts throughout the testing process
Phase (3)	Observation	▪ To observe the interactions of users throughout the testing process ▪ To detail additional usability issues
Phase (4)	Survey based Questionnaire	▪ To establish the degree of user satisfaction with the learning resource ▪ To collect user recommendations and devise solutions

The sample contained 10 participants, all of whom had had some degree of participation in the designing development process of the dEv. There is vast argument pertaining to the usability evaluation sample size and how many participants should be included; thus, numerous authors have come to recognise that usability evaluation does not require a large sample size. Therefore, our sample size was aligned with the suggestion made by Nielsen in regards usability evaluation concept, and further aligns with the 10±2 rule (Hwang & Salvendy, 2010; Nielsen, 2000) for various reasons. First and foremost, this is the first

evaluation session for the dEv model, which provides the fundamental elements for creating the dEv model. Moreover, there also are no significant functions that require a large assessment sample. Secondly, as the literature mentioned, a large number of participants for usability evaluation would mean the repeating of the same issues already identified in a small sample of participants, which is costly. Thus, the recruited sample size was aligned with the purpose of the study.

The main criteria for the sample selection were as follows: subjects should have programming experience; and the subjects should have a minimum of one designing product and be able to learn more about the user interaction design assessment approaches. This work has carried out in-person sessions, involving individuals sitting with the researcher on a one-to-one basis. Approximately 1 hour was assigned to each participant. The subjects were seen to have some degree of dEv-use experience, albeit differing: 3 subjects described themselves as 'beginners', whilst 7 were 'intermediate' and 1 'expert'

Procedure

Each participant took approximately one hour to complete this study tasks and collect feedback. At the beginning of the session, the researcher introduced the study aims and procedure, and then asked them to sign paperwork. This study asked participants performing two tasks to achieve the study goals (See appendix 1:B.2). These tasks aimed to preamble participants on the learning resource. There were two test scenarios, including four test cases, which were to be completed and would take approximately 20 minutes. Afterwards, participants were asked to choose any unknown evaluation method form the dEv resource and free testing to explore and learn from it. During the free-testing, some issues were highlighted for discussion with the participants and enabled further collection of feedback and usability problems. Furthermore, thinking-aloud conducted during the free testing part to collect more data about the participant finding issues. At the end of the session, the questionnaire was filled in in an effort to measure user satisfaction with the resource (See appendix 1: B.1).

7.3.3 Study Findings

This section's aim is to provide an overview of the evidence garnered from the key findings of the literature, as well as from the empirical works supporting the decisions made by system novice developers and their overall involvement in the earlier stages of system design;

improving usability assessment capacity and the garnering of end-user feedback and user requirements.

System novice developers' decisions and their involvement in the early phase of the design system

Usability Themes

An overall this study comes up with 9 usability main themes, which can be divided into 26 sub-themes. The main themes and sub-themes discussed during the users testing sessions. Table 7-8 shows the main themes and sub-themes discussed during the usability testing sessions. Table 7-9 shows the list of new themes identification and interpretation during the user testing sessions.

Table 7-8: Main themes and subtheme descriptions and research finding interpretations

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
video	Clear visualisation	Videos have a specific structure that integrates the clips and short summaries of the video. Participants said that 'video structure is clear and contains important information that well described the video'. All the participants agreed that structure is clear and understandable, especially before running the clip to understand the video content	The complete agreement of the video structure interface design means that participants were happy to have a short description about the video contain. As discussed, this description helps them to understand what this video is about, and shows that participants were happy to have a combination between videos and short text description of the video. The literature review has highlighted the substantial role of assistant tools used for increasing user's knowledge about related topic (Aberg & Shahmehri, 2000).
	Links to jump	Some of the included videos have long clips that could be undesirable as a whole; however, participants suggested that links to jump are better in order to avoid unwanted clip parts: for instance <2(2:25- 4.00) put link on 2:25 that let the user to jump at this time. Three participants mentioned the same issue.	All participants were interested in watching short videos rather than long videos. Participants mentioned three different techniques to improve the providing videos, which are cutting video, links to jump and directly running a specific part of the video. These different techniques clearly showed that participants have different levels of experience and viewpoints. However, all of these suggestions

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
	Cut the clip	Long videos containing unwanted durations should be cut to show only the required time. Participants said 'cut the video and make it smaller'. Four participants supported cutting the videos in order to show only the required clip.	clearly show that long videos were not of interest and should not be included because nobody would be willing to watch them. Short videos with relevant and concise information reflect on the quality of the user experience. Though long videos might contain all the information needed, it might distract the user from the surrounding environment (Boiano, Bowen, & Gaia, 2012).
	Run specific section	Long video that contains unwanted sections should be included however let the video automatically running the required time then stop. Stop reminder is required to include that remind the user stop the clip and keep the time appear on the full screen. Two participants mentioned that a stop reminder should be included.	
Top Menu Style	Modern	There was only one participant who mentioned having a top menu instead of a left menu.	This study shows the two opposing suggestions of the menu style. The first group supported the top style whilst the second

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
	Not appear all time	Participants mentioned that the top menu would not be available all the time, which will annoy users.	group supported the left menu style. Participants interested in seeing the top style was because of its modern style and its common use for interface design. However, the second group argued that the top menu would not appear all the time on the screen, meaning this will increase user actions on the interface. Furthermore, the top menu could distract the user with the browser tool bar, which may allow the user to leave the resource. Moreover, the top menu might not be clear enough for the user against the left menu style. For these reasons the majority of the participants supported the left menu for this resource. Menu style design is considered to be significant in making information on web sites easy to find. Although previous research has suggested a left menu is preferred by users, recent literature also claim that the selections of menu style is a personal choice as long as it is usable (Burrell & Sodan, 2006). Essentially, all of these two suggestions of menu style are
	Distract	Top menus will disturb browsing windows, meaning users frequently attempt to click on the browser bar rather than the main menu bar.	
Left Menu Style	Clear and understandable	Eight out of ten participants agreed that the left menu is clearer and more understandable. However, one participant suggested a list of links rather than a menu as it would be more stylish.	
	Naming	The menu section naming needs to be more clarified and should better describe the topic. Four participants supported this issue whilst one made the following suggestion: 'What is it?' Instead of what? 'How to use it?' Instead of	

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		how .	commonly used; however, the tradition left menu style will fit on this learning resource, as supported by participants. Furthermore, there were some suggestions on the menu items naming, meaning participants were interested in improving the menu and keeping it as the appropriate style of the learning resource.
References	Distract	Reference on the main home page is distracting to the user.	Including references in the design content (text, image, video etc.) is important to give software content more credibility and allow application users to build up trust for the contents. This technique is commonly used on web applications: for example Wikipedia provides a list of references on the same page as the content. There are conflicts pertaining to Wikipedia's accuracy of information; however, Kräenbring *et al*. (2014) suggest that Wikipedia is an accurate and complete source of study (Kräenbring et al., 2014). The study software placed the references as a part of the main
	No need	Some users do not need the references to appear all the time because they don't need for them. Three participants highlighted this proposal.	
	Too much data	References mean the home page loads too much data on the screen. Users are required to focus on important data only. Three participants supported this suggestion.	
	Easy to find	References placed on the main page make it easier to	

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		find references if needed. Four participants supported this issue.	interface. This included the reference of the chosen topic only. There were two groups against each other: the first group supported the view that references should not be included as a part of on the main interface; the second group supported putting references with some changes to the structure. Each group came up with some reasons as to why their choice was supported. Participants who suggested leaving the reference on the main interface mentioned that is easy way to find further information without any extra effort. However, the second group mentioned that references should be somewhere out of the main interface because it's too much data on the screen, which will distract the user from reading the important parts. Furthermore, these references are not requested all the time, meaning it is appropriate to move them somewhere else.
	Deduce user effort	Participants supported the right references because this reduces users' efforts and actions on the interface.	
Colours	Black and grey	Black and grey interface colours are supported by three participants, who mentioned that using	The design features, such as colours, are referred to as a factor that enhances software usability (J. H. Song & Zinkhan, 2003).

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		simple and limited colours gives more clarity. In addition, these two colours are formal and commonly used, which can be read form any devices and formats.	The study participants came up with two different views on the interface and text colours. The first view supported the simple and formal way of using black and grey colours only. These participants who supported this suggestion intended to be more simple and clear. The second group supported the use of multiple colours on the interface and mentioned the interactive design as the main factor for keeping users dealing with the interface design. It is true that designers want to allow the user to come back and use it again; however, consistency is important in caring when completing the design. The next version uses colour with various changes in order to be more consistent. Moreover, it is possible to have another version including only black and grey.
	Colourful	Four participants were interested in having a colourful interface rather than black and grey (simple). They mentioned that using more colour is attractive to the reader. However, designers should be careful with colour choices and be consistent.	
Long text	Collapse and expand text	Long text is not suitable; thus, using the collapse and expand method is important. Four participant supported this technique because this method makes	Scrolling is the default method to read or see long text on a browser. However, the study shows that some participants support short text on the interface by using the collapse and expand method '+,-'.

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		the interface clearer and allows the reader to focus on the important content, which also encourages the reader to explore and read the rest of the content. One participant supported this technique; on the other side, however, he mentioned the need to 'keep the important information and hide the unnecessary information that can disturb users'.	Those participants wanted to have a clear interface design and be presented with only the important information, with an additional option to expand the information. Furthermore, this was also mentioned as unnecessary information only. All of them agreed that some information should be hidden. In this study, there were three participants who argued that all including information is important to read and should appear all the time; otherwise, some users would not read some sections or would get lost somewhere. Johnson argued various advantages and disadvantages of using scrolling on the design (T. Johnson, 2013). The combination between these two techniques is expected to be more useful and will improve usability; however, consideration is given to which information will be hidden and which information will be appear.
	Scrolling	Three participants mentioned that information should be directly appearing on the screen because this is important information and it cannot be hidden. Furthermore, the scrolling technique will reduce user actions on the screen.	
Images	Reduce long text	Using images that describe the long text is suggested by participants, especially	Image, charts and graphic are essential for quickly learning a complex topic. Images summarise

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		for describing workflow.	long text or workflow. The study participants mentioned images need to be considered because they present a lot of information in a simple and short way, meaning users can quickly scan and images are a way of having an interactive design. Some images are included but more will be added on the next version.
	Clarity	Images will make the interface clearer and the information more readable.	
	Time	Images should be supported because these methods will reduce the time of learning.	
Home Page and its related pages	Content	The main page provided a short sentence of the topic and each sentence described on separate page. Four participants support the view that each main page information should start the same on its own page. Furthermore, there were some spelling mistakes found.	The consistency of information between the main pages and its related pages are important in order to clearly present the topic. The main page contains a short sentence 'hint' for the topic and the user has the control to explore the topic using menu or links. Therefore, some participants mentioned that the title of the main page should be changed to clearly present a summary page for the topic. Thus, change is considered a high priority and should be updated to improve the level of usability and stop distracting the reader. Cook & Dupras (2004) listed ten guidelines for creating an educational website, suggesting the design of pages should include
	Title	Each topic has presented on one main page (home page) and it's related to many other pages that explore the topic. Three participants were supported that main page title should be changed and call it 'Summary' because this	

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		will avoid confusing readers.	clear headings, sentences comprising no more than 20 words with a maximum of 5 sentences on each paragraph. Furthermore, each topic should start with a summary before giving more details (Cook & Dupras, 2004).
Navigation	links and its locations	Participants stated that 'main page links are stated on the perfect place which is at the end of each section'. This was supported by another two participants. Moreover, participants suggested other links for next and pervious pages at the end of each page, which allows the user to control the pages without using the main menu. One participant mentioned that links are an important way of jumping between topics; however, this could mean the user loses where they are. Thus, designers should find a method for keeping the user informed of the current place: for instance,	The difference in navigation methods is important to consider on the interface design as this gives users multiple ways of controlling the interface. Therefore, links to navigations have been used on the study designing interface and mostly are accepted by participants. The study shows the link locations were at the right and clear place. However, extra links for more navigation were mentioned as provided: for instance, 'next' and 'previous' at the end of each page. This means participants could deal with multiple navigation methods or do this through their own design experience. The user getting lost could be the main cause of preventing the use of links for page navigation. Thus,

Main Theme	Sub-theme	Findings from the study	Research Interpretation for the finding
		'highlighting the new topic'	'breadcrumb' navigation could be a solution for this (Magazine, 2009).
	Links to jump	Four participants supported the jump up links placed at the end of each section on the screen as this will reduce the user's action from scrolling up.	
	broken links	There were some broken links found for both internal and external links.	It's possible to have some broken links for technical problems. Thus, some broken links found for both internal and external links, meaning the participants carefully tested each single point on the interface.

7.3.3.1 Enhance the usability evaluation behaviour

The present work has identified various new themes, as highlighted by the study subjects. Table 7-9shows the list of new themes identification and interpretation during the user testing sessions.

Table 7-9: New themes identification and interpretation

New identified themes	**Interpretation of the themes identified**
Lack of Information	This feedback is clearly shows that some participants are interested in having extra information and additional topics, also its way increase their awareness of decision making(Alkhuraiji, Liu, Oderanti, & Megicks, 2015). The resource should targets both novice and expert users, thus including and advance topic should be planned and take it as a separate study. Researcher needs to review the topics and meet users in deciding how to integrate this on the main resource.
Contact E-mail	This is known as utility navigation feature and is considered to be one of the activities strongly impacting user satisfaction with the design (Farrell, 2015). This suggestion is one of the most important feedbacks. This suggestion means the resource updating will be regularly and based of the users using. We must keep on touch with users all the time and create an email or contact form for any further suggestions and feedbacks.
References Design Solution	This evaluation study allows users to be involved in the design process by creating suggestion to reinstruct the design. A references section is one of the sections where participants are given some examples for redesign. This means some users are willing to be involved in the design process by giving design suggestions. These suggestions are considered on the next version of the software.
Text and Reference Integration	The integration between the contents and references is important to keep the user related with the original sources for the content. This method also will reduce the time of learning about references between the lists of references. However, this integrating could be way to distract the users with a lot of references links. The searching tool could be a solution instead of the integration, thus we should planning to add this service and well presented on the further version of the resource.

7.3.3.2 Gathering of end-users' satisfaction and feedback

At the end of the user testing sessions, participants were asked to complete a user satisfaction questionnaire. Table 7-10 shows the participants' agreed percentages in regard to the dEv interface elements. This study show that participants were in complete agreement that dEv has clear structure, easy menu style, enough content to understand the topic and the images that provided are helpful too. However, 70% of the study participants agreed that dEv provides an easy navigation. Extra information about each topic is included as references, where these references have been placed as the part of the topic main pages. In total, 70% of the participants rated these references as helpful references, which encouraged them to explore the topic in depth; in contrast, 10% disagreed and 20% were undecided. However, only 60% agreed that references should be placed at the right position on the interface whilst 10% of them disagreed. The study results show that 80% of the participants agreed that the links provided were clear and easy to find; however, 20% of them rated this as unclear. Using videos on the dEv resource were rated as a useful way of understanding the topic by 80% of the study participants whilst the rest 20% remained undecided. However, 60% of the study participants agreed that the videos provided were helpful and reduced learning time, whereas 10% were disagreed and 30% were undecided in terms of whether or not these videos were helpful and the right choice (See Appendix 1: B.3 dev user satisfaction result).

Table 7-10: Percentage of agreed statements

Statements	% Agree
Clear interface structure	100%
Easy navigation	70.0%
Menu Style	100%
References position	60.0%
Content structure helps me to clearly understand the presented topic	100%
Using links through text to jump between different topics is obvious	80.0%
References encourage me to expand the topic for more information	70.0%
Using images is helpful to understand the topic	100%
Using videos is helpful to understand the topic	80.0%
The videos provided reduced learning time	60.0%

Overall satisfaction is an important goal when applying the questionnaire. The participants were asked to rate their overall satisfaction with three elements: the design of the software interface, the appearance and the usability. The results show that, overall, 90% of the study

participants were satisfied with the design of the software interface whilst 10% were neutral. The software appearance was rated as satisfied by 80% of the participant; however, 10% of the participants were dissatisfied, whilst the same percentage were neutral. The majority of participants (90%) were satisfied with the usability level of the dEv software; the rest (10%) were dissatisfied and claim it should be improved (see Table 7-11).

Table 7-11: An overall satisfaction with dEv resource

	Dissatisfied	**Neutral**	**Satisfied**
Software Interface	0 (0.0%)	1 (10.0%)	9 (90.0%)
Appearance	1 (10.0%)	1 (10.0%)	8 (80.0%)
Usability	1 (10.0%)	0 (0.0%)	9 (90.0%)

7.3.4 Discussion

This study was conducted in order to evaluate the first version of the dEv resource and accordingly come up with a new improvement planning. The study data has been collected by using three data collection methods: user testing, short interview and questionnaire. This result has been categorised to 9 usability main themes, which can be divided into 26 sub-themes. Table 7-8 presents all of these usability themes; the research has interpreted the research findings. Most of the users testing issues have produced based on two against groups that argued and came up with two different suggestions. However, all participants agreed about the interface elements. Participants made suggestions relating to the interface elements style only: for instance, the menu style and video structure. This study provides a list of new themes suggestions, as shown in Table 7-8.

The dEv resource has been built based on multimedia (videos with words and images) and links (either navigation or references links). Thus, some usability problems or improvement suggestions are expected from the study participants. The following list provides more discussion on the study themes and subthemes.

Video

As shown in the results section, participants have an interest in everything being as short as possible, with most of them making suggestions or comments on the long text and videos. This is a valuable finding that tells us about the future planning of the software development and has led us to care more about the short-term. Videos are an essential method for reducing learning time because they work with multiple senses. Thus, Mayer (2009) mentions that learning can be more effective if there is a mix between words and pictures, such as in the

form of videos (Mayer, 2009). This study learning resource combines text and description for each video. The study participants were happy with the structure and considered it a good video summary. The assistant tools are also helpful in increasing user knowledge on the related topics (Aberg & Shahmehri, 2000). The learning cognitive load is valuable and should be minimised on the learning resource. Mayer & Moreno (2010) state that 'cognitive load is a central consideration in the design of multimedia instruction'; thus, learning cognitive load should be reduced and they summarise number of ways that can be solving for this challenge and reduce the cognitive overload (Mayer & Moreno, 2010).

All videos provided present the original resource without any changes. The reason for keeping these videos without change is measuring user satisfaction with existing videos and how these are accepted by participants. However, the study results show participants were not completely satisfied with the length of videos. Thus, they came up with three different solutions to reduce the video time. These solutions are as follows:

1- Cutting the videos and keep only the required clip time that required to see

2- Links to jump into the required section that controlled by users

3- Automatically running the required clip time only and give the users opportunity to see the rest of video if they are interested.

These three suggestions clearly show that most of the study participants were unhappy with long videos, with short videos more acceptable. Based on the participants' suggestions, the videos will be reduced using one of the previous user's suggested methods. Furthermore, whole videos will be added for participants who are interested in seeing the whole video instead of reading. Thus, these videos will be placed on the 'further resource' section for each topic of the resource.

Menu

Menu navigation styles are one of the best methods for presenting web information through an easy approach (Burrell & Sodan, 2006). Thus, many different styles, such as (add common menu style) menus, are commonly uses on the web applications design. The study participants were divided into two groups, with each group supporting a different menu style. The first group supports the top-menu style whilst the second group stayed with the left-menu style. The first group state the top menu style as being modern and commonly used; however, the second group with 8 participants argued for a number of reasons. The decision was made to stay with the left menu in order to address the top men style. This difference is expected,

with Burrell & Sodan (2006) stating that menu style is a personal choice (Burrell & Sodan, 2006). Maguir (1999), in his study, listed a number of recommendations for creating a menu style on software applications: firstly, short options (items) can be easy to understand and use, meaning menu items should be no longer than 12 options (items) and need to be clearly named without any abbreviations; secondly, the menu list and order should be grouped alphabetically order (M. C. Maguire, 1999).

Based on the study results, the left menu style is appropriate for a learning resource and is easy to understand. Therefore, the next version on this learning resource will keep the left menu style and will add various improvements.

References and Citations

Credibility and trust are two important factors where learning resources should be met to allow the user to continue with the information. References and citation are two methods that can allow the learning resources information to be more credible and trust. The literature review mentions that providing citations and references on web design is a significant helpful contributor to a web design credibility (Fogg et al., 2001). The dEv learning resource is concerned with these two factors in mind of encouraging users to use the product as a learning resource. Thus, the references have been placed as part of the main interface for each topic. During the user testing, participants were in complete agreement about the usefulness of including these references on the learning resource; however, they argued about their position. Based on the study results, there were two groups of references up for argument: on the main interfaces or moved from the main interface as a folder on the main menu under each topic. It was believed that this movement would mean users would not be distracted by too much data. Both groups had good reasons supporting their choice; however, the majority of the participants supported moving them from the main interface; thus, the next version will remove references from the main interface and include them as a menu item. This could mean they are easier to find and are more attractive to users.

Interface Colours

Based on the study data collection, there were two groups of participant against each other in the use of colours on the interface design. The group supported the use of many colours whilst the second group supported the use of simple and formal colours (black and grey). Using colours can be both positive and negative for the software usability. Thus, the literature argued these two concepts of using colourful and simple colours on the interface design. Many studies supported colourful interface with the limited use of colour. Magguire (1999) states that colourful displays provide an attractive interface; thus, the use of multicolours is

recommended. However, there are a limited number of colours that should be included on the design (M. C. Maguire, 1999). Web content accessibility guideline mentions 'do not rely on colour alone', meaning the design content, text and graphics should be clearly understandable and viewed without colour for any reason (Chisholm, Vanderheiden, & Jacobs, 2001). Coloured test is preferable on the menu; however, neutral colours, such as black or white, are recommended (M. C. Maguire, 1999). On this study, there was not much difference between the number of each group suggestions, meaning the use of colour with a limited range was decided on as positively impacting users and keeping them focused on the topics and contents.

Long Text

The interface length of content is a factor that this study attempts to measure, as well as the length of text can impact the using of the learning resource. As a result of this study, two groups were found to be concerned with the length of the text, and they came up with two different techniques for this issue. The first technique was supported using the collapse and expand method '+,-'and the second technique supported the scrolling method. Each technique has advantages and drawbacks. Bevan (1999) states that web site users are willing to scroll web pages if required, although the inclusion of navigation links or buttons will reduce the need for scrolling (Bevan, 1999). However, Peytchev *et al*. (2006) conducted a study that measured effectiveness on users' responses of using both pages and scrolling survey design. Their study found that scrolling design takes a long time from users to finish the survey because the interface design affects them. Furthermore, long scrolling periods led users to miss a lot of questions (Peytchev, Couper, McCabe, & Crawford, 2006). Johnson argued that the content collapsing technique has a number of advantages: for example, it allow a lot of the interface information to be compressed, hides long texts and allows the user to choose their method of interaction. However, there remain usability problems after using the content collapsing technique: for example, searching services cannot present highlighted words placed on the collapse sections (T. Johnson, 2013). This issue is debated, and there is no specific best choice because it is a personal choice and comes down to preference. Thus, the combination of these two techniques is expected to be more useful and will improve usability level; however, this considers which information will be hidden and which information will appear.

Images

Images are important to simplify and present the complex information as opposed to large volumes of text. Images should be created or included to be close to reality as possible.

Developer needs to be consistent with involving images and graphics because too many images could be distracting or confusing to users. Furthermore, images that require users to scroll to see the whole image should be avoided, with images resized for users (M. C. Maguire, 1999). The dEv learning resource provides a few workflows images, whilst study participants are interested in seeing more images that support learning resource topics. This means participants consider images and graphics as solutions able to reduce content through an attractive approach.

Home Pages and Related Pages Structure

Design content is another important factor in design. Each design is required to ensure good interaction with users. The interaction is produced by two elements: attentive design and providing content. Thus, balance between these two elements is not easy (Huizingh, 2000). The paragraphs should not be long (maximum 5 sentences) and each sentence should be not more than 20 words, although a short summary of the page topic is required to include at the beginning of the page(Cook & Dupras, 2004). Rosen & Purinton (2004) cite that web content is one of the main factors impacting the number of repeat users visiting the website; thus, developers should more care about the page's contents and presentation (Rosen & Purinton, 2004). Another study has shown that high-quality content which is easy to use and the frequency of updating is the main reason behind repeating the visiting of the web. Page titles are also important for design navigation. Each page on the design should have a specific title as titles are significant and helpful for users to establish their location on the design; using the same title for numerous pages can be confusing to users and cause them to feel lost (M. C. Maguire, 1999). Based on the study results, summary pages (main page of each topic) should be titled 'SUMMARY' to let users know this is the summary page for the whole topic.

Links Navigation

Including the number of features on the web site, such as navigation bar and hyperlinks, gives users a freedom to browse the web by jumping to different sections without backtracking. However, unbalance in these features could lead users to experience difficulties in remembering their movements (Mohd Sam & Tahir, 2009). A hypertext technique is considered a positive opportunity for creating intelligent content that interacts and affects different levels of users' understanding (Maroney, 1997). Links give users more control on the interface, which Maguire (1999) mentions: design should include a clear navigation to help users control the design(M. C. Maguire, 1999). The study learning resource includes a number of hyperlinks to jump between topics and different locations on the page. Participants were happy with the link locations and names; however, they suggest adding more links to

move from one topic to another, such as 'next' and 'previous', at the end of each page. Nielsen & Tahir (2001) mention that providing links should be clear, understandable and use short names as users usually scan the first and second words on the links. Clear explanations about what the links contain is required, with terms such as 'click here' or 'more…' to be avoided. Instead users should be told exactly what they will get more of, such as 'More Book Reviews'(Nielsen & Tahir, 2001). It would be useful to give different colours for the visited and unvisited links as this increases search efficiency and will reduce the users losing their position(Halverson & Hornof, 2004). Based on the study results, participants were happy with recent link locations and names, and were willing to deal with different navigation methods to reduce a loss of time, such as 'breadcrumb' navigation, which could be a solution and help to ensure users know their locations all the time.

New themes Suggestions

During the user testing sessions, a number of recommendations were listed by participants to improve their learning resources, as shows in Table 7-9. These user recommendations will be considered for the next resource development process, which are:

1- Expanding information: some participants said that provided information is so lack and needs to expand and make it appropriated for both novices and experts users.

2- Video and reference structure redesign: some participants have come up with number of suggestion design to improve the presenting of both videos and references sections.

3- Provide utility navigation feature: providing contact email to let the users contact the resource novice developers for any reporting or future suggestion after deploying the resource.

4- Content and references integration: make the content provided related to the original references, which is one way of letting users check or expand on topics more easily.

7.3.5 Conclusion and Outcomes

The key element in the design of software is usability, with usability improvement also fundamental in the development process. Novice developers undergoing training on UEMS is essential so as to ensure their decision-making concerning software usability is encouraged. Nonetheless, creating such learning resources is not a simple task. In this work, we examined the effects of learning resources (dEv) on software usability, as well as on the usability

decisions of developers. Accordingly, the resource was designed in line with particular specifications garnered throughout prior works. This study carried out various evaluation methods in an effort to gather the study data.

This study provides a key contribution concerning the research area's knowledge by creating a learning resource that encourages novice developers in the application of UEMS and to enhance the decision-making of usability amongst novice developers. The study findings emphasizes that eh learning resource influences software novice developers on the general usability perspective. More specifically, novice developers are well positioned to complete usability assessments on their products and make choices in regard to their overall usability. Learning resources that are designed in line with particular requirements may have an influence on UEMS importance and usability understanding. Furthermore, the study results emphasis the fact that the involvement of users in the earlier phases are fundamental when seeking to ensure the usability of the software is improved. The study findings are important for the interested researchers in terms of developing understanding that novice developers really are in need of prompting and applying UEMS by themselves, and are able to make decisions on usability. This will lead researchers to work hard in this area in order to increase novice developers' overall ability to create usable software. Furthermore, the study findings also are important for system designers as this can lead them to be concerned about software evaluation concepts and increase their confidently in evaluating their products so as to achieve usable design.

One of the study's main limitations is the fact that 10 participants in the UK were responsible for the empirical data. Such a small sample therefore means the findings cannot be generalised; therefore, subsequent works should make use of a larger sample of subjects in different industries so as to establish a more in-depth learning resource that is able to promote UEMS-related understanding amongst novice developers, in addition to their overall decision-making capacity.

summary

Chapter Eight: New Implemented Approach-dEv Framework

This chapter has detailed the creation of the dEv framework and how the various dEv integration methods are associated with the framework stage. The framework's challenges and impacts are also discussed in this chapter.

8 New Implemented Approach-dEv Framework

8.1 Introduction

A number of studies provide evidence to show that the integration of agile and users is being improved upon so as to ensure the creation of usable products. With this in mind, the present chapter details a new model created in an effort to improve usable software creation. The new framework is referenced as the Design Evaluation framework, labelled as the dEv framework. This model has been designed in line with the incorporation of two concepts, namely user involvement and iterative. Various works were carried out in mind of integrating agile with another software development concept; therefore, such studies are regarded as creating new dEv frameworks. The new suggested model is focused on being at the centre of the development process, and devises development stages. Moreover, the approach provides various quick evaluation methods geared towards impacting all development stages. The figure 8-1details the new model's high level.

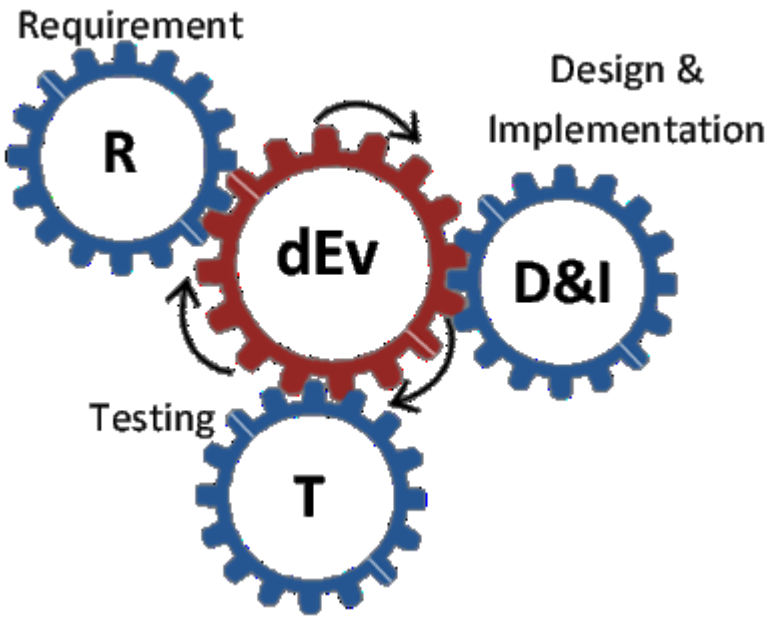


Figure 8-1: The high level of the dEv framework

8.2 Framework Phases

The framework has been designed centred on UCD and iterative concepts; therefore, the framework's individual stages have been detailed so as to ensure the key stages of the iterative model have been covered. The iterative model, as detailed in the above figure, emphasises the four key phases in each iteration in the implementation of the final version of the product. These are detailed as follows:

1. Planning in mind of the individual requirements
2. Implementation-related analysis and design.
3. Testing
4. Evaluation.

In actuality, the new dEv framework is seen to comprise the first three phases, namely requirements, design/implementation and testing, as these are recognised as the underpinning phases in any software development model. Furthermore, the number of integrated evaluation methods is linked with each individual phase, recognised as the dEv methods phase.

The subsequent sections provide greater understanding on each individual stage and its workings.

Requirements

The requirements phase is the first in the development process, during which all requirements need to be established and examined. Moreover, all requirements, whether function or non-function, should be established (Bassil, 2012; Sommerville, 2010). Users, experts and organisations alike should gather various requirements, whether business, software or user, for example (Wiegers & Beatty, 2013). No particular methods are necessary for gathering the requirements; however, different forms of data collection approach are considered suitable in the gathering of requirements, such as questionnaire and focus group requirements (Courage & Baxter, 2005).

Design and Implementation

During this stage, a preliminary design should be created in line with the requirements gathered. The product architecture will be determined in this stage so as to ensure the design's individual components are designed. User and evaluation methods need to be adopted so as to gather the data related to design improvement. Throughout this stage, the finalised version of the design should be coded and applied through the utilisation of implementation language; this helps to establish the product.

Testing

Throughout the testing phase, the design needs to test and measure via applications centred on testing and establishing both technical issues and any problems facing the user (Whittaker, 2000). Testing in this vein means 'the process of executing a program with the intent of finding errors' and to make sure that the program works as it should (Myers, Sandler & Badgett, 2011). Sound evaluation planning is necessary and can help to ensure evaluation costs are minimised (Perry, 2006). The adoption of various evaluation methods can be useful in assessing software quality and usability, which is fundamental when seeking to enhance software (van Velsen, van der Geest, Klaassen & Steehouder, 2008). As can be seen

inFigure **8-2**, there are various suggestions deriving from the data analysis testing; these can affect all other stages. Accordingly, the recommendations feed both the implementation and design stages, and all requirements on all iterations.

8.3 dEv Framework Integration Methods

This stage is recognised as the main one in the model as it links all other stages and has a direct impact on all. This link between the various stages and methods provides a list of methods that can be adopted in order to facilitate developer reacting usable software. Moreover, is seeks to encourage novice developers to adopt evaluation methods across all phases. Each of the approaches applied have been determined in line with inexpensive and cheap evaluation methods so as to ensure novice developers are encouraged to learn and carry out evaluations. Discount usability is regarded as one of the most important terms for quick and inexpensive e valuation, as determined by Nielsen (Nielsen, 1994) in the Heuristics Evaluation (HE), with Task Scenario and Thinking Aloud, also another two common evaluation methods (Focus groups and Questionnaire) added. As discussed earlier, the creation of valuable software is the key objective to establishing a model; thus, as noted by ISO, there are three factors that need to be measured so as to make decisions relating to software usability, namely efficiency, effectiveness and satisfaction. Such methods have been afforded to three groups, each of which is linked to particular framework stages.

Two different methods have been included in the requirements stage, namely focus group and questionnaire, which are recommended due to their ability to gather requirements relating to users and software. The focus group is valuable in gathering a wealth of qualitative data through the completion of group discussions and data analysis so as to devise a list of requirements (Gill, Stewart, Treasure & Chadwick, 2008). The questionnaire method also may be applied in mind of examining a large population, with the data analysed so as to establish the requirements. The questionnaire method has been applied in mind of gathering quantitative data (G. Marshall, 2005).

The second phase inherent in the design and application of the dEv framework provides two methods, namely Heuristics Evaluation (HE) and Thinking Aloud. The evaluation of the design may be carried out through implementing HE at the preliminary design stages, even if not only on paper (Nielsen, 1995). The Thinking Aloud method also may be carried out

throughout the process of performing HE, with evaluators asked to speak aloud when completing the design check. This approach provides user interaction and user behaviour feedback (van Velsen *et al*., 2008).

The final stage in the dEv framework is testing, with five evaluation methods carried out. In this stage, the solution needs to be assessed, meaning all evaluation methods should be applied in mind of measuring specific factors. For example, the questionnaire may be carried out so as to gather insight into user satisfaction with the product. This questionnaire may be devised by the researcher or through the adoption of existing questionnaires, such as SUS. Moreover, scenarios also may be designed and users asked to perform them with the Talking Aloud method so as to assess the product's efficiency and overall effectiveness. Both the user and expect can use the software created and measure it against the heuristics of Nielsen.

The creation of a resource that can help framework users to understand the evaluation methods provided is fundamental. Accordingly, the dEv resource should be created so as to teach novice developers about such methods and how these should be completed. The learning resource is pivotal in helping novice developers with a lack of knowledge pertaining to the evaluation methods. Accordingly, the dEv learning resource is one of the subsequent results.

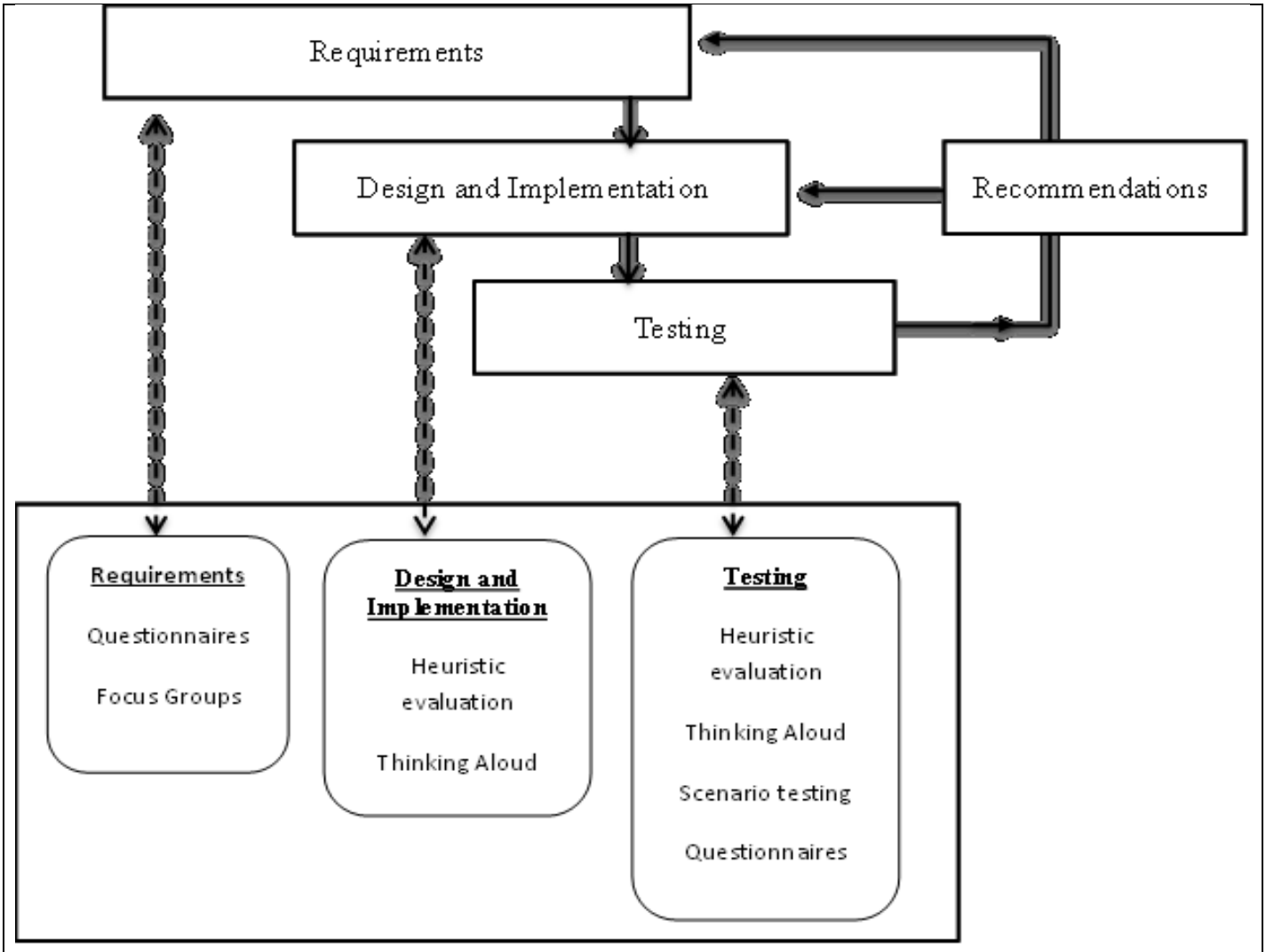


Figure 8-2: The dEv framework for software development

8.4 dEv Framework Impact

Moreover, it seeks to circumvent any degree of miscommunication between designers and developers. Lastly, the method supports the lack of quick and inexpensive integration evaluation approaches.

User Involvement

The dEv framework has been designed in order to integrate the development process stages with various evaluation approaches so as to ensure user involvement is enhanced throughout the course of the development process. As can be seen in the figure, each stage may be linked

with the group of methods necessary to carry out the collected data and thus enhance software. Accordingly, the dEv framework is a valuable way of encouraging novice developers to consider users and how they can be involved.

Developer Ability for Conducting Evaluation

Accordingly, the various methods detailed on the dEv model are necessary for learning resources to teach novice developers how such methods can be carried out in order to improve their overall ability to complete evaluation methods and also to expand knowledge in this regard. The resource has been devised and explained in further detail in Chapters 9 and 10.

Design Principles

Software development principles, as implemented by developers, should be wide-ranging and should cover different topics as opposed to only particular topics; principles need to teach developers as opposed to telling.

In sum, the suggested framework is focused on achieving user involvement, software usability and software evaluation concept integration so as to create usable interaction design. Moreover, developers also are taught how to confirm usability through the use of different methods and principles.

8.5 dEv Framework Challenges

Resource creation that motivates novice developers in how evaluations how can carried out is one of the challenges of this study. Past work has shown that developers experience challenges in carrying out usability testing (Salah, Paige & Cairns, 2014). With this noted, teaching the performance of evaluation by developers is a challenge. Various authors have noted that online learning context design is challenging for designers; this means that creating a combination of activities that can satisfy various elements, including the needs of students and learning objectives (Anderson, 2004). Moreover, the wide-ranging implementation of the internet and its associated tools mean the creation of resources is more problematic and requires careful design; therefore, online learning resource problems are identified as a challenge affecting learning (L. Song, Singleton, Hill, & Koh, 2004). Accordingly, this highlights the prompting of novice developers in the use of learning resources to affect

various elements as a challenge in the present work. Evaluating the entire impact of the model and learning resource also is identified as a challenge as it requires a long-term assessment in order to devise a comprehensive assessment model. Therefore, the assessment carried out in the present work is just a part of a whole assessment.

8.6 Final dEv Tool Description

The dEv resource is fundamentally built upon the core concepts of discount usability, as specified by Nielsen in chapter 3 section 3.2.3. In addition to the core concepts adopted by Nielson in discount usability, dEv also supports other efficient and easy-to-use methods, namely the use of questionnaires and focus groups. The dEv resource covers four main techniques for evaluating software:

- Focus groups
- Heuristics
- Questionnaires
- Scenarios/Thinking aloud.

8.6.1 Design and Development Process of dEv

The development methodology adopted was UCD integrated with agile software development. Thus, the project development can be divided into three phases: Requirement Gathering, Software Design and Implementation and Testing.

- **Requirements Gathering**

Throughout this phase, the developer collected the Functional and Non-Functional requirements of a support tool. These were gathered using several focus groups sessions with the following objectives:

  - To achieve a better understanding of the support tool and its purpose.
  - To establish the scope of the support tool structure and contents that need to be included.
  - To determine current and future releases.

- **Software Design and Implementation**

In this phase, the developer designed the initial Graphical User Interface (GUI) and database design. As with all agile development, several iterations were required in order to ensure the user interfaces more user-friendly.

For example, adopting the familiar tree view representation as the following representation:

- Folder icon for the main pages. 
- Document icon for the subpages. 

In this phase, the developer began implementation. The steps used were as follows:

- Implement a page at the client side
- Implement a page at the server side
- Implement the functions necessary to connect the database using the Data Access Layer (DAL)
- Test the current page
- Repeat the above steps for all pages.

- **Testing**

In this phase, the novice developers test the entire support tool, including all pages (interfaces), with the use of multiple test scenarios. Usability evaluation sessions were used to verify that all requirements had been completed correctly. Furthermore, in this phase, the developer received various suggestions for future development of the tool following the usability evaluation with users. Subsequently, the developer classified this feedback and profited it using two levels: low and high. The low level means the changes should be done, but the high level means changes must be done.

8.6.2 Technologies Used to build dEv Resources

The development technologies used in the project can be divided into sections:

- **Database:** Microsoft SQL Server was used as the website database. Furthermore, all connections were made through the stored procedure in order to satisfy the following objectives:
  - To improve security of the web
  - To reuse code in different places

- To improve performance overall.
- To ensure maintainability.
- To achieve network bandwidth conservation.

- **Website:** The website uses multiple technologies, including the following:
  - The main language is the ASP.Net, which was coded using C#
  - Bootstrap, which is a framework for developing responsive websites
  - JavaScript and Cascading Style Sheets (CSS)
  - TinyMCE which is a platform independent free text editor released as open source under LGPL.

8.6.3 dEv Interfaces Structure

The dEv resource interface is made up of two main sections, namely the menu and the content sections (see Figure 8-3).

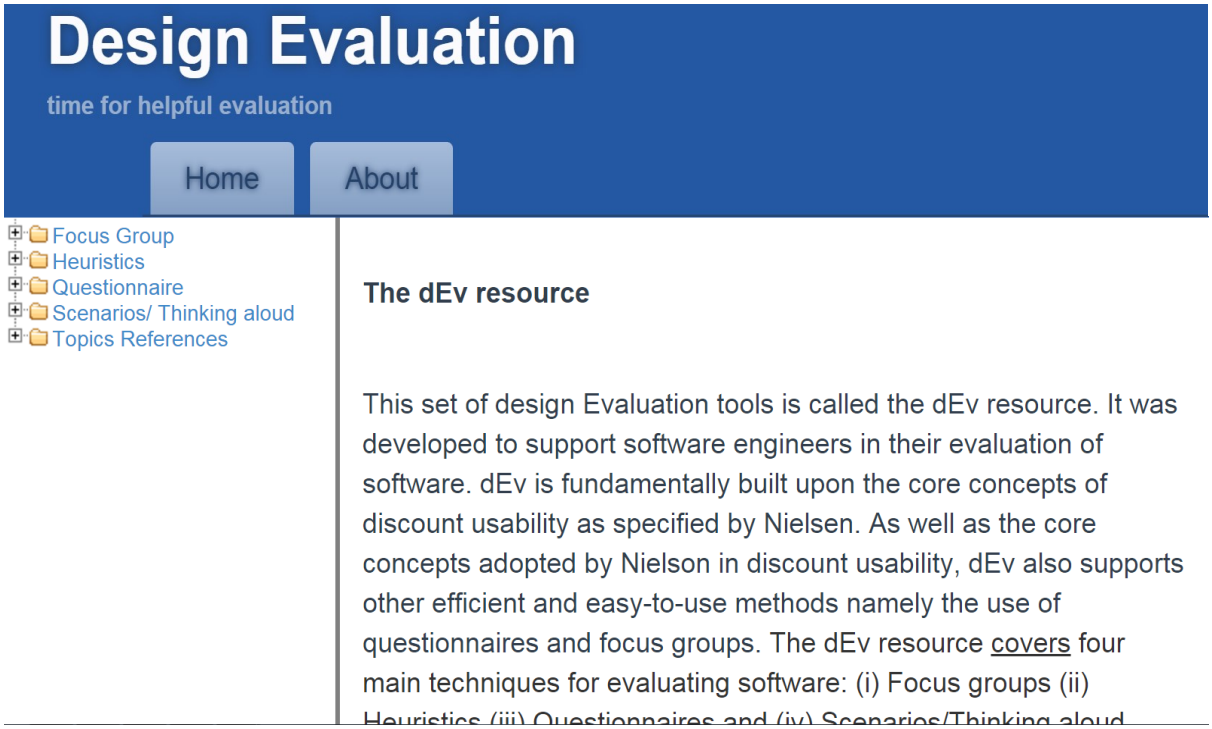


Figure 8-3: dEv Main Interface

Menu Section:

This area of the interface presents the main menu for the dEv resource. The tree menu style was used owing to the fact that it is a traditional and easy style with which most computer

users have experiences. This menu provides expand and collapse functions to make the tool navigation easier and more understandable for end users. Furthermore, each folder icon represents one of the main topics supported by the tool in more details. In the requirement gathering focus groups sessions, users made it clear they required a tool which was early to navigate. Furthermore, they requested a simple, consistent structure which enabled them to quickly find topic of interest. As the developer worked with the content, is become clear that a structure which focused on the keywords What, Why, Where and How could work well. Therefore, each folder contains at least four files (see Figure 8-4), which are:

- What: this page provides a definition of the topic (method) and further explanation about it
- Why/Where: this page provides a reason/several reasons for the use of the topic (method) and where it is appropriate to use
- How: this page provides a deeper explanation of the topic (method) and how it can be applied in order to achieve useful feedback
- Further resource: Providing more resources and references that enable users to read more about the topic (method).

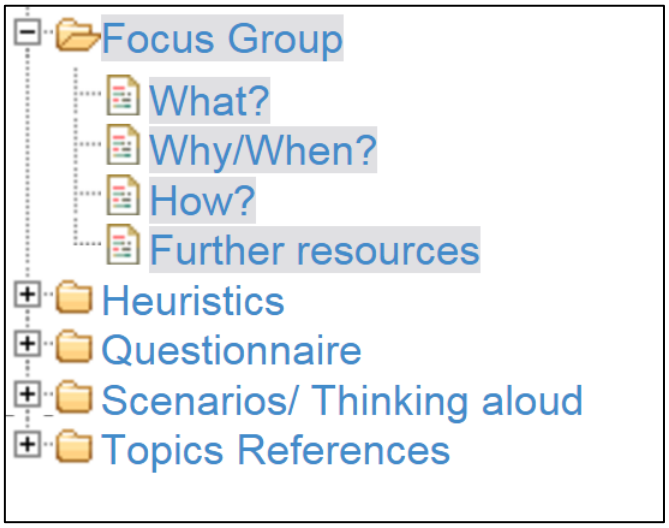


Figure 8-4: dEv Menu Style

Main Body:

When the user selects a page from the menu, the page is shown in the centre of the main body of the display. The contents of the page combines: texts, videos and workflow (images) (see Figure 8-5

).

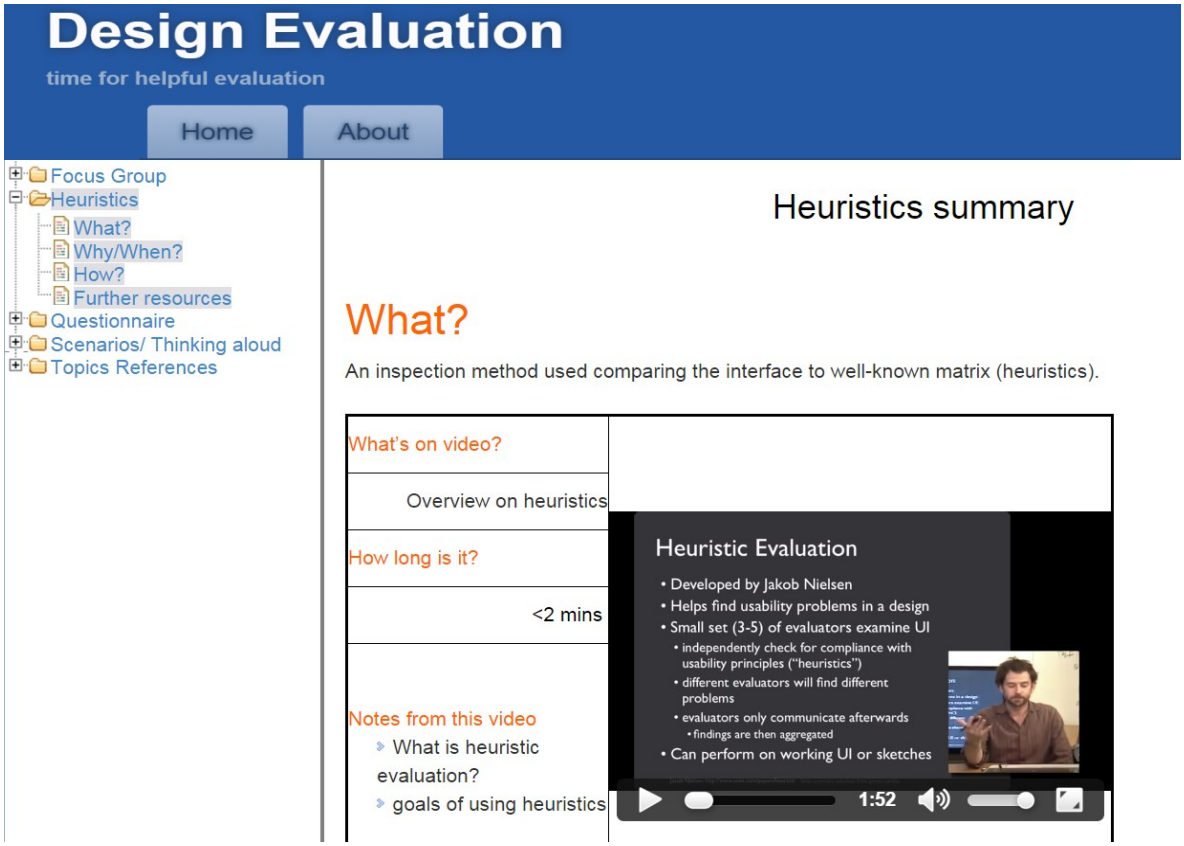


Figure 8-5: The Main Body Section

The dEv learning resource is already available, please visit the link below
<< http://devresource4u.com/Default.aspx >>

8.7 Summary

Through this chapter, the integration of iterative along with evaluation methods has been witnessed, with the model designed recognised as encouraging software engineers to implement design evaluation throughout the process of software development. The framework has been designed in line with the key concepts of discount usability, as determined by Nielsen (Kheterpal, 2002; Nielsen, 1993a). Moreover, the dEv framework also includes two other easy-to-use and effective methods, i.e. focus group and questionnaire.

The design of the framework has emphasised how evaluation methods may support each phase inherent in the process of software development (Almansour & Stuart, 2014). Overall, this model recognised the dEv approaches as critical to the development process and as

pivotal in assisting the software development phase. Importantly, the framework impacts novice developers to ensure the involvement of users throughout the development process in the preliminary stages of the process, and also increases the ability of the developer to carry out evaluation sessions. Moreover, through the adoption of different evaluation methods, many design principles have been applied so as to measure usability as opposed to merely following a list of principles. As with any study, challenges were faced; these have been taken into account.

Chapter Nine: Empirical Evaluation of dEv Tool Effects on Design Activity

This chapter details an experiment concerned with the empirical evaluation of the dEv model and the effects of the learning resource from the perspective of the design developers. Two groups performed the Shneiderman and dEv design frameworks, whilst the third group focused on the developer framework. The comparison work was completed on the basis of these three groups, concerned with establishing the number of involved users, the amount of acquired changing behivour from all individual design frameworks, and the evaluation methods applied. The learning resource and dEv model were concerned with enhancing the perspective of developers in involving methods and users throughout the development process, centred on achieving a usable solution.

9 Empirical Evaluation of dEv Tool Effects on Design Activity

9.1 Study Motivation

Ensuring access to the suggested dEv framework is pivotal in order to establish both the differences and similarities with other methods. There was the creation, assessment and improvement of the dEv design framework throughout prior works. Notably, this is the first experiment to have been completed with the application of the dEv design framework and learning resource as a required approach to product creation. Importantly, this work seeks to evaluate the overall efficiency of the model as a learning tool/instrument when teaching users interface design methods; and secondly, it contrasts those experimental data garnered for software novice developers with the use of the dEv tool as learning resource, amongst other approaches. Three individual developer groups were used, working with various design methods. The contrast between the three developer groups will be considered in line with three different concepts: first, the number of different methods of assessment applied; second, the amount of users involved; and third, the volume of acquired changing behaviour from each of the respective design approaches. The experimental data were examined and analysed with the aim of establishing the effects of the development framework, guidelines, tools or learning resource on the behaviour of the developer when completing product creation.

9.2 Study Methodology

In order to satisfy the study aims, the experimental methodology will be utilised. The present work has adopted three different methods of data collection where each one is centred on the fulfilment of particular objectives. Moreover, the gathering of data through the application of various methods will be pivotal in gathering information valuable to the comparison study. These different strategies of data collection are experiment, interview and questionnaire. The first method completed is shown in Table 9-0-1.

An interview data collection method is one of the best methods for gathering in-depth face-to-face information. Novice developers were invited to an individual interview for approximately 50 minutes. The interview aimed at identifying the developer's strengths and weaknesses, and the difficulties faced during the software development. The interview also identified how the design method helped them to complete the task. These interviews were

valuable in helping the researcher to understand the development process followed in the creation of software, and how the design method affected the process. An online questionnaire also was adopted and novice developers were required to fill in at the end of the experiment. This questionnaire aimed at capturing information regarding the implemented solutions and the developer's own satisfaction with the design method employed. Questionnaires are recognised as valuable when striving to gain insight into users' feelings throughout testing and accordingly measuring their degree of product satisfaction (Bargas-Avila et al., 2009). The questionnaire is regarded as a suitable method for gathering quantitative data to compile statistics (Holzinger, 2005).

Table 9-0-1: Research method phases for the first validation dEv experiment

Phase No	Phase denomination	Purpose and achievement
Phase (1)	Experiment	▪ Establishing access to the dEv model's overall efficiency and that of its learning resource. ▪ To facilitate the completion of a comparison study
Phase (2)	Interview	▪ To establish the various strengths and weaknesses of the developer, in addition to the obstacles encountered throughout the software development process. ▪ To establish the way in which the design method proved valuable in the completion of the task.
Phase (3)	Questionnaire	▪ To measure the satisfaction of the participant in regard to the design approach implemented.

The sample contained 12 participants. The main criteria for the sample selection were subjects should have software programing experience at any level. Participants were randomly divided into three groups of four novice developers. This research

Participations Groups

These groups of participants were randomly allocated without criteria. This randomly grouping was applied to make sure that final result are valid and avoid any bias (Odgaard-Jensen, Vist, & Timmer, 2011). Furthermore, researcher was using three techniques to prevent any effect on the final results. Firstly, participants did not have any idea about the investigator and the required topic. Thus, all participants had the equal time to deal with the task. Secondly, there was one of these groups called dEv group which this group members were required to follow our suggestion tool; thus this group members did not know that this required framework was our new suggestion. Thirdly, participants could have a different level of programing skills, which could impact the overall rate of the group solutions; thus the implementation stage was not required, in order to avoid any differences in the technical skills.

The experiment participants were divided into three groups as follows:

1. dEv group: this group was required to use the dEv framework and learning resource(ref for website link)
2. Shneiderman's group: this group was required to use Shneiderman's Eight Golden Rules of interface designing (Table 9-3)
3. Control group: this group was required to use their own design methods/experience.

All of the subject participants were male undergraduate students, with various programming experience. The demographic data for the study subjects is detailed in Table 9-2.

Table 9-2: Demographic descriptive statistics for the first dEv experiment group of subjects

Group	Gender	Age	Experience	Occupation
dEv	M = 4 F = 0	1 = 2 2 = 2 3 = 0	≤1 = 0 2 = 2 3 = 1 4 = 1	UG = 4
Shneiderman	M = 4 F = 0	1 = 4 2 = 0 3 = 0	≤1 = 1 2 = 2 3 = 1 4 =0	UG = 4

Control	M = 4 F = 0	1 = 2 2 = 1 3 = 1	≤1 = 0 2 = 2 3 = 0 4 = 2	UG = 4
Values = variables mean values M = Male F = Female Gender: M (Male), F (Female)} Age: 1 (18–24 years), 2 (25–34 years), 3 (35–44 years) Programing experience: ≤1 (year or less), 2 (two years), 3(three year), 4(more than three years) Occupation: UG (Undergraduate)				

This division aimed to gather different information to produce the comparison study between these three groups. The following section shows how the methodology applied and conducted.

Task

The participants were asked to produce an application that could present an exam room clock. The clock would be a project installed on a wall during an examination so that students could see the time remaining. This particular activity was selected owing to the fact all of the individuals involved in the study were students and had come across this particular issue in the examination room. Moreover, they were recognised as having a suitable environment in which data pertaining to their product requirements, design and evaluation could be garnered. Participants had the freedom to use any programming language for task implementation; however, each participant was required to use a specific design method to complete the task.

Procedure

The novice developers were asked to complete the task by following a specific designing method. The task process passed through three different stages:

1. Task deploymen.t
2. Task implementation.
3. Software submission and feedback collection.

The first group was assigned to completing the task using the dEv learning resource to guide the creation of the user interface. Thereafter, this group will be referred to as the dEv

development group. The dEv learning resource is an online resource that novice developers were able to access at any time during the task creation. The second group was assigned to completing the task using Shneiderman's Eight Golden Rule to guide the creation of the user interface. These eight principles considered as one of the famous design principles toward usable software(Shneiderman et al., 2013). Thereafter, this group will be referred to as the Shneiderman's Development Group. A copy of Shneiderman's research paper containing his Eight Golden Rules was given to this group (See Table 9-3). The third group was assigned to complete the task using their experience to guide the creation of the user interface. Thereafter, this group will be referred to as the Control Development Group. These novice developers were free to choose any method of developing the software interface, based on personal preference and experience.

The experiment participants needed to learn the design development method applied to the creation of the software interface. The participants groups were allocated assigned two weeks to complete the development of the software and submit a copy of the task solution with short report about the creation process. The researcher has carried out the last stage of the experiment process, which is centred on collecting novice developers' feedback. An interview was conducted with each participant to identify more details about their development strategies for creating the task solution (See appendix 1:C.1). At the end of the interview sessions, participants were asked to complete an online questionnaire. The questionnaire contained 11 questions and was divided into two parts: the first part contained demographic and background questions; the second part contained questions relating to the requirements of the collection methods employed, as well as the number of users involved in the development and the novice developers' likes/dislikes of the design method used(See appendix 1:C.2). Finally, this process generated the data to be analysed by the researcher to produce the study result. In line with the data derived from the questionnaire and interview, there was data analysis in order to establish the study results.

Table 9-3: The Eight Golden Rules of Interface Design (Shneiderman, 2015)

Principles	Description

Principles	Description
Strive for consistency	Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent color, layout, capitalization, fonts, and so on should be employed throughout. Exceptions, such as required confirmation of the delete command or no echoing of passwords, should be comprehensible and limited in number.
Cater to universal usability	Recognize the needs of diverse users and design for plasticity, facilitating transformation of content. Novice to expert differences, age ranges, disabilities, and technological diversity each enrich the spectrum of requirements that guides design. Adding features for novices, such as explanations, and features for experts, such as shortcuts and faster pacing, can enrich the interface design and improve perceived system quality.
Offer informative feedback	For every user action, there should be system feedback. For frequent and minor actions, the response can be modest, whereas for infrequent and major actions, the response should be more substantial. Visual presentation of the objects of interest provides a convenient environment for showing changes explicitly.
Design dialogs to yield closure.	Sequences of actions should be organized into groups with a beginning, middle, and end. Informative feedback at the completion of a group of actions gives operators the satisfaction of accomplishment, a sense of relief, a signal to drop contingency plans from their minds, and an indicator to prepare for the next group of actions. For example, e-commerce web sites move users from selecting products to the checkout, ending with a clear confirmation page that completes the transaction.
Prevent errors	As much as possible, design the system such that users cannot make serious errors; for example, gray out menu items that are not appropriate and do not allow alphabetic characters in numeric entry fields. If a user makes an error, the interface should detect the error and offer simple, constructive, and specific instructions for recovery. For example, users should not have to retype an entire name-address form if they enter an invalid zip code, but rather should be guided to repair only the faulty part. Erroneous actions should leave the system state unchanged, or the interface should give instructions about restoring the state.

Principles	Description
Permit easy reversal of actions	As much as possible, actions should be reversible. This feature relieves anxiety, since the user knows that errors can be undone, and encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data-entry task, or a complete group of actions, such as entry of a name-address block.
Support internal locus of control	Experienced users strongly desire the sense that they are in charge of the interface and that the interface responds to their actions. They don't want surprises or changes in familiar behaviour, and they are annoyed by tedious data-entry sequences, difficulty in obtaining necessary information, and inability to produce their desired result.
Reduce short-term memory load	Humans' limited capacity for information processing in short-term memory (the rule of thumb is that we can remember "seven plus or minus two chunks" of information) requires that designers avoid interfaces in which users must remember information from one screen and then use that information on another screen. It means that cell phones should not require re-entry of phone numbers, web-site locations should remain visible, multiple-page displays should be consolidated, and sufficient training time should be allotted for complex sequences of actions.

9.3 Study Findings

The study results have been produced based on the novice developers' solutions and feedback. This data has been analysed as a comparison study carried out amongst these three groups of developer. The comparison study focused on three important elements, namely:

1. The number of users' involved
2. The number of design evaluation methods used, and
3. The amount of acquired changing behaviour of designing from each design principle.

First of all, it is important to identify how the novice developers have considered the software design from the early stage of the software development. The researcher asked the novice developers about their first thought on the software development after they received the task. Data collection shows that novice developers have adopted one or both of the two techniques to design their software:

1. User requirements (who adopted the strategy for gathering user requirements for the task)

2. Their own experience (who made use of their own experience to devise a solution to the problem without consideration to user requirements).

Number of Users Involved

Questionnaire was applied to collect the number of user's involvement and the novice developers were asked to identify how many users have involved during their designing process. Thus, designing principles could be a way of encouraging novice developers to involve users during the software development process. The dEv framework was presented to novice developers as a guide centred on gathering insight into requirements, software design, and implementation and testing. The framework encourages the participation of users in each stage; thus, the dEv development group has included users at least twice as often as the Shneiderman and control development groups. There were more than 17 users involved by the dEv group, 8 users by the control group and 3 users by the Shneiderman group.

It also important to identify which stage of the software development process included the highest number of users between the development groups. Data collection shows that all the dEv novice developers asked users to provide software requirements prior to designing solutions. Each developer involved a minimum of two users. This can be contrasted with the Shneiderman group, where one developer asked one user to define requirements and the control group that did not gather requirements from any users. At the design stage, three of the dEv novice developers involved more than 8 users. The Shneiderman and control groups failed to involve any users. The testing stage saw the greatest involvement of users by all novice developers groups. Each group asked at least one user to test the software by using either evaluation methods or informal testing that tried breaking the system. The dEv novice developers involved 14 or more users, which is the highest number of users involved by any development group. This can be contrasted with the control group that involved 8 users whilst the Shneiderman group involved 2 users in mind of testing the software by one developer. Table 9-4 shows the number of users involved according to each group at each development stage. In total, the dEv development group registered the highest number of users involved amongst the Shneiderman and control groups. This means the dEv framework has clearly identified that user involvement is the main key to using the dEv framwork.

Table 9-4: The number of users involved in the requirement, design and testing development stages for each development group

Frequency of the number of user involved in the Requirements stage				
		dEv	**Shneiderman**	**Control**
Number of users	No Users	0	3	4
	1 Users	0	1	0
	2 Users	1	0	0
	3 Users	2	0	0
	More than 3 Users	1	0	0
Frequency number of user involved in the Design stage				
		dEv	**Shneiderman**	**Control**
Number of users	No Users	1	4	4
	1 Users	0	0	0
	2 Users	1	0	0
	3 Users	1	0	0
	More than 3 Users	1	0	0
Frequency number of user involved in the Testing stage				
		dEv	**Shneiderman**	**Control**
Number of users	No Users	0	3	0
	1 Users	0	0	1
	2 Users	0	1	2
	3 Users	2	0	1
	More than 3 Users	2	0	0

The Number of Design Evaluation Methods Used

The number of evaluation methods used during the software development is essential to improving the quality and usability of the software (van Velsen et al., 2008). The design framework can encourage software engineers to apply a variety of design evaluation methods. Table 9-5 shows the evaluation methods and the number of times each was used by a development group. Eight methods were identified based on the results of the online novice developers' questionnaire and the data collection conducted throughout the course of interviews.

The eight (8) evaluation methods were applied a total of 20 times across all groups. The dEv group performed 13 evaluations with the use of seven of the eight techniques. This accounts for 65% of the evaluations performed. The Shneiderman group performed 4 evaluations using only one technique; this accounts for 20% of the evaluation performed. The control group performed 3 evaluations using two techniques. This accounts for 15% of the evaluations performed. The dEv developer included a wide range of evaluations techniques, whilst the control and Shneiderman approaches remained focused on a limited set of evaluation methods. The control group employed only the user testing and task scenarios whilst the Shneiderman group remained focused on the Shneiderman eight principles only. These results illustrate the impacts of the design framework on developing usable software. The Shneiderman group failed to consider any techniques outside the framework, essentially considering the framework to be a complete definition of usable software. The control group, which had an overarching framework, performed only the most limited usability testing as it was considered good practice. Only the dEv group selected a broad range of evaluation techniques as the framework emphasised this approach to the development of the usable software.

Table 9-5: The usage of evaluation technique by the development group

Evaluation methods that have used	**Number of using**			**Total**
	dEv	Shneiderman's	Control	
Questionnaire	3	0	0	3
Interview	1	0	0	1
Task analysis	3	0	1	4
Thinking aloud	1	0	0	1
Focus group	2	0	0	2
User testing	2	0	2	4
Heuristic evaluation	1	0	0	1
8 golden rules	0	4	0	4
Total	*13(65%)*	*4(20%)*	*3(15%)*	*20(100%)*

The Amount of Changing Behaviour from Each Design Principle

Both dEv and Shneiderman novice developers were asked about their view of using the design method and how this new tool changes their design behaviour. Data collection shows that three out of four Shneiderman novice developers listed these principles as design principles only; these novice developers also said that there is no mention of any evaluation methods for application side-by-side with these principles; that means the Eight Golden Rules cannot be a guide for design evaluation and are for interface design only. However, 50% of dEv novice developers have considered that the dEv framework is a guide focused on how to use the evaluation methods during the software development process. On the other hand, 50% of them considered this framework a combination of design principles and design evaluation methods, meaning that novice developers can use the framework as a guide to creating and evaluating software.

These variations in novice developers' views concerning the goal of the design framework clearly showed that developers have the ability to explore the purpose behind the design framework. Thus, the design framework can be a way of teaching developers more about the design principles and design evaluation methods (Bruun & Stage, 2014; Howarth et al., 2009; Skov & Stage, 2005) . Data collection shows that novice developers found their individual design framework to be a usable guide for aiding task completion. However, each developer saw different advantages in their framework. Firstly, the dEv framework and learning resource promote software engineers to use the design evaluation methods by themselves. Data collection shows that the dEv novice developers have learnt some of the design evaluation methods and successfully applied them to their software solutions. Novice developers liked the dEv framework way of learning the evaluation methods in the early stage of their designing experience. According to dEv participant, software engineers have enough information about implementation tasks, but poorly deal with users and software evaluations methods. The dEv framework taught the software engineers about finding system weaknesses and failures at any stage of the software development process. The combination of design principles and the evaluation methods was one thing that novice developers mentioned as improving their experience of the software development: for example one developer can be quoting saying 'The 10 principles gives you a checklist to work against, before you take this to a focus group. I probably saved myself a lot of time redesigning by following these methods'. The dEv framework helped the novice developers to apply the design evaluation methods as one developer said, 'The ability to be able to apply the method

to the application. It was very easy to look at what I was developing and what questions to ask users based on the application and the method'. Secondly, the Shneiderman's novice developers have learnt a lot from their method; however, their learning concept is quite different from dEv novice developers. Secondly, the Shneiderman's novice developers behaviour was focused on the design principles more so than the design evaluation methods. This was expected because these principles are focused on how to design the interface. According to one participant, the Shneiderman's principles 'help to create the interface design by telling me what principle the design should focus on, such as reducing short term memory load'. The novice developers have not tried to use additional evaluation methods except the Eight Principles, and there is no mention of involving users in the software development process. Importantly, this question was not posed to the control group as they have used their own experience designing approach. Moreover, the principles underpinning the Shneiderman and dEv models were not used.

Development Stage

The subjects were asked to establish both the difficult and most valuable development phase from their perspective. Data collection recognises that the requirement phase was rated by subjects 8 different times as being important; the testing and implementation phases were rated 3 times each, whilst 2 ratings were assigned to the design stage. Moreover, the requirement stage was rated 6 times as a difficult stage, whilst 3 ratings were recognised for the testing stage. A total of 2 ratings were afforded to the implementation stage and 1 rating to the design stage (see Table 9-6).

Table 9-6: Stage Ratings

	Important stage	**Number of rating times**
	Requirement	8
	Design	2
	implementation	3
	Testing	3
Difficult stage		

Requirement	6
Design	1
implementation	2
Testing	3

9.4 Discussion and Conclusion

The key objective of this experiment was centred on establishing how the design methods implemented impact the designing perspective of the novice developers. It was apparent in the experiment results that the participants had enough implementation knowledge to create such as task. Accordingly, all the study participants completed the creation of a software solution for the given task. The participants used their coding experience and implementation language of choice for the solution creation. This experiment was not looking at the types of language used to complete the task or how complex the given task was; rather, it focused on the design method used to create the solution. Thus, the design framework used by each group differentiates the solution using two aspects, namely:

1. Number of users involved
2. The number of design evaluation methods utilised.

The study results show that the dEv development group is more confident about their solution. Their confidence comes from the design framework used to create the task. This led them to perform various design evaluation methods that increased users' involvement during the software development process. Thus, the dEv novice developers moved from being programmer-focused on coding solutions to software engineers who focused on the overall design process and the usability of the software system. The dEv software engineers group learnt how to involve users during the software development process, and also how to evaluate their applications without external help. In line with the findings, a total of 2 of 4 different dEv solutions were designed in line with user requirements. In contrast, however, solutions amongst the control group and Schneiderman subjects were devised in consideration to only their experience. Thus, evaluation learning resources are a way of increasing the usability and quality of the software by teaching software engineers about design evaluation (Brézillon, Borges, Pino, & Pomerol, 2008). A comprehensive evaluation is required for improving the usability and quality of the software interfaces. The functionally

testing is not enough to improve the software quality and usability; however, the software effectiveness, efficiency and user satisfaction with software is important. This study shows that new software engineers are able to conduct evaluation sessions and effectively deal with users to improve the software. The dEv novice developers provide the evidence that new developers have the ability to apply design evaluation methods. The dEv development group used 65% of the available evaluation methods and involved 76% of the total number of users involved by all study groups.

The study findings support the view that novice software developers have the ability to learn usability assessment approaches and to implement such approaches individually. Moreover, the designing process of the dEv group was clearly able to demonstrate the positive effects of the design methods on the design strategy applied by the novice developers. For example, in total, 13 evaluation methods were applied by the dEv group during the course of their design process; in contrast, however, 7 evaluation methods were applied by the control and Shneiderman groups. This significant difference is seen to relate to the design method performance.

In conclusion, this study comes up with a list of recommendations to improve software usability. These are:

1. Design frameworks affect novice developers, and where they emphasise quality and usability they will encourage novice developers to follow the guidance.

2. Therefore, they should adopt a framework that includes these principles and methods of design evaluation.

3. The logical phase to introducing such teaching is after the developer has learnt to code (learnt to become a programmer), such as when they are beginning to study software design on the road to become software engineers (usually Year 2 of an undergraduate degree).

The result should be a software engineer that produces usable software in a cost-effective manner.

Chapter Ten: Empirical Evaluation of dEv Tool Impacts on End-User Satisfaction

This chapter provides an experiment study that applied the dEv model as supporting resource for developers. The study developers were divided into two groups: the first was given the dEv learning resource as a support for software evaluation; the second group was not given any support. The final applications were evaluated and ranked according to the System Usability Scale (SUS). The results showed the dEv learning resource strongly impacts the usability of the final applications. Furthermore, research found that undergraduate software engineers are willing to conduct some of the evaluation methods under expert supervision.

10 Empirical Evaluation of dEv Tool Impacts on End-User Satisfaction

10.1 Study Motivation

Previous work has developed a software development tool, namely the dEv framework, which aimed at teaching the principles of usable software development. This study investigated how software novice developers' use of software evaluation methods during the development process impacts product user satisfaction level. A group of second-year students was recruited. For the first time, these students were undertaking a project where they were required to complete the analysis, design, implementation and testing of a software solution. This project is known as the integrated project as it pools together all of these software development areas. The study novice developers were divided into two groups: the dEv group was provided with the dEv framework as a supporting tool, whilst the non-dEv group was asked to complete the project using human computer (HC) knowledge taught on their degree programme. The students' completed projects were collected and evaluated by testing users; testing users were asked to complete the System Usability Scale (SUS) questionnaire so as to provide an objective means of satisfaction with the end product.

10.2 Study Methodology

This study applied an experimental methodology to achieve its aims behind performing the study. This work has implemented four data collection methods, where each method aimed at achieving specific objectives. Moreover, the gathering of data through the application of various methods will be essential in gathering information valuable to the comparison study. These different strategies of data collection are performing tasks, user test, interview and questionnaire. The research method phases are shown in Table 10-1.

First of all, the study novice developers were required to complete the module-required tasks. Based on the completed tasks, the following data collection methods will be performed for gathering the study data. User testing was adopted in order to test the developer's applications by testing users. Dealing with real applications helped the testing users to rate their satisfaction with applications. Questionnaires are recognised as valuable when striving to gain insight into users' feelings throughout testing and accordingly measuring their degree of product satisfaction (Bargas-Avila et al., 2009). Thus, the System Usability Scale (SUS)

was applied in order to collect user satisfaction with completed applications during the usability evaluation sessions (see chapter 6). Comparison study between applications in term of usability is aimed of this study, thus SUS is appropriated questionnaire to make comparison between different applications (Brooke, 2013). SUS considered as reliable and valid, also can be applied to different of technology (Sauro, 2011a). See Appendix 1: D.1 for SUS survey. An interview data collection method is method for gathering in-depth face-to-face information. Novice developers who completed the task were invited to an individual interview for approximately 50 minutes. The interview aimed at identifying the developer's strengths and weaknesses, and the difficulties faced during the software development. The interview also identified how the design method helped them to complete the task. These interviews were valuable in helping the researcher to understand the development process followed in the creation of software, and how the design method affected the process (See Appendix 1: C.1). An online questionnaire also was adopted and novice developers were required to fill in at the end of the interview sessions. This questionnaire aimed at capturing information regarding the implemented solutions and the developer's own satisfaction with the dEv model employed. Both SPSS and NVivo analysis software tools were applied in examination of qualitative and quantitative data.

Table 10-1: Research method phases for the study

Phase No.	Phase Denomination	Participants	Purpose and Achievement
Phase (1)	Task	Novice developers	▪ To create the required tasks for the module
	User testing	Test users	▪ To assess the web and desktop applications
Phase (2)	Interview	Novice developers	▪ To collect novice developers feedback of the designing process and dEv model
	Survey based Questionnaire	Test users Novice developers	▪ To establish the degree of user satisfaction with applications ▪ To collect user recommendations

Participants

This study involved two types of participant, namely novice developers and general testing users.

Novice developers

The sample contained five novice developers (4 males, 1 female) were involved as novice developers. The main criteria for the novice developers sample selection was involved in the integrated project module for the second-year computer science course at the Plymouth University. Each participant submitted a project with a Web application and a Desktop Application. The novice developers fell into different age groups, with 40% of them in the 18–24 years age group, whilst the same percentage was identified as falling into the 25–34 years age group, whereas the remaining 20% were aged 35–44 years old. All the novice developers have at least two years' programing experience.

Test users

The sample contained a total of 58 testing users, evaluating the twenty software applications (ten applications developed by study novice developers and another ten collected from the course leader for another five novice developers for the evaluation purpose only). Users generating a total of 300 SUS questionnaires. The main criterion for the testing Users sample selection was familiarity with computer and web applications. These users only aimed to evaluate the developer's applications. Figure 1 shows the testing users demographic data across the 58 users.

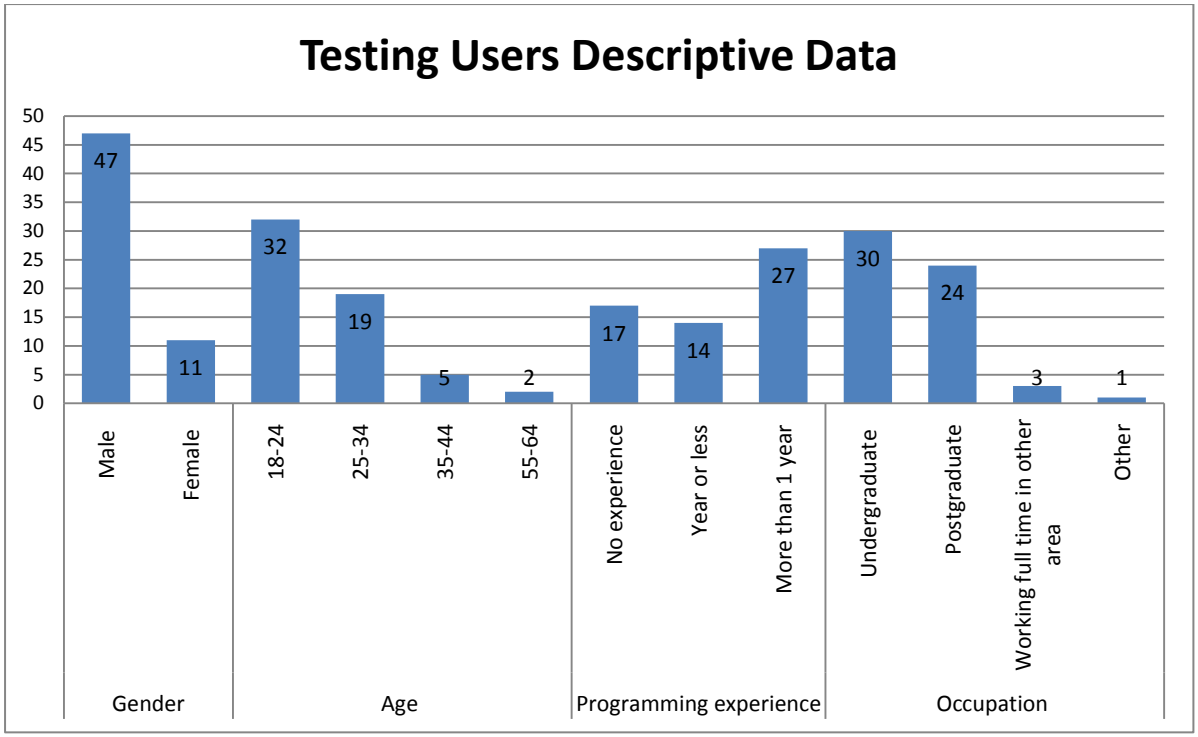


Figure 10-1: Representing the study testing users demographic data

Demographical data for each group testing users

Group A:

The group applications were tested by 30 users (28 males, 2 females). The users were seen to hold two different degrees of education (16 undergraduate; 14 postgraduate). Users tested the application fifteen times each. There were 18 users with more than one year programing experience, 8 users with one year or less, and 4 users with no experience at all. There were two age groups involved: 18–24 years (20 participants) and 25–34 years (10 users).

Group B:

This group created Web and Desktop applications, each of which was tested 15 times in an effort to derive user satisfaction value. In total, there were 30 participates (25 males, 5 females) in the users satisfaction lab. These were divided into four age groups:

1. 18–24 years (13 users)
2. 25–34 years (11 users)
3. 35–44 years (2 users)
4. 55–64 years (4 users).

There were 12 users with more than a year programing experience, 10 users with a year or less, and 8 users with no experience at all. There were three occupation categories held by users: undergraduate (14 users), postgraduate (12 users) and full-time employees (4 users).

Group C:

Table 10-2 shows Group C's Web and Desktop applications were evaluated by 28 male and 3 female users. These users tested the applications 30 times (Web 15 times, Desktop 15 times) in order to gather information into user satisfaction values. There were 19 users with more than one year programming experience, 4 users with a year or less, and 7 users with no programming experience at all. In total, the applications were tested by 13 undergraduate users, 15 postgraduate users and two full-time employees. There were three age groups involved: 18–24 years (13 users), 25–34 years (14 users) and 35–44 years (3 users).

Group D:

This group of Web and Desktop applications was tested 15 times with the involvement of 26 male and 4 female users. There were three different occupations held by the users: undergraduate (17 users), postgraduate (12 users) and full-time employee (1 participant). There were three age groups involved: 18–24 years (16 users), 25–34 years (10 users) and 35–44 years (4 users). There were 10 users with no programing experience, 8 users with a year or less, and 12 users with more than one year's programming experience.

Group E:

Group E Web and Desktop applications were evaluated 15 times with the involvement of thirty (30) users (20 males, 10 females). Users' ages ranged between 18 and 44 years; 18 were aged 18–24 years, 10 were aged 25–34 years and 2 were aged 35–44 years. Users had different levels of programming experience: no experience at all (14 users), one year or less (8 users) and more than one year's experience (8 users). There were four occupations degrees held by the users:

1. Undergraduate (14 users)
2. Postgraduate (12 users)
3. Full-time employee (2 users)
4. Other occupation (2 users).

Group F:

Table 10-2 shows that F group Web and Desktop applications were evaluated by 30 users (24 males, 6 females). Most were undergraduate students (20 users), whilst 8 of them were postgraduates and the rest 2 were full-time employees. There were 10 users involved in each

programming experience level. There were three age groups involved: 18–24 years (18 users), 25–34 years (10 users) and 35–44 years (2 users).

Group G:

Group G Web and Desktop applications were evaluated 15 times with the involvement of 27 males and 3 females. Users had two occupations degrees: undergraduate students (18 users) and postgraduate students (12 users). In total, 4 users had no programming experience, 11 users had one year or less, and 15 users had more than one year's experience. There were four age groups involved:

1. 18–24 years (15 users)
2. 25–34 years (12 users)
3. 35–44 years (1 participant)
4. 55–64 years (2 users).

Group H:

The web and Desktop applications were evaluated by 23 male and 7 female users in order to establish their overall satisfaction with them. In total, the applications were evaluated by 19 undergraduate students and 11 postgraduate students. There were two age groups involved: 18–24 years (22 users) and 25–34 years (8 users). There were 6 users with a year or less of programming experience, 8 users with more than one year's experience, and 6 users with no experience.

Group I:

Group I's Web and Desktop applications were evaluated 15 times by 26 male and 4 female users. There were three age groups involved: 18–24 years (22 users), 25–34 years (6 users) and 35–44 years (2 users). Users were either undergraduate students (22 users) or postgraduate students (8 users). Users held different levels of programming experience: no experience at all (10 users), one year or less (6 users) and more than a year's experience (14 users).

Group J:

There were 30 users (24 males, 6 females) that participated in evaluating the Web and Desktop applications. Users held different levels of programming experience: 7 users had no experience at all, 6 users had one year or less, and 17 users had more than one year's experience. There were four age groups involved: 18–24 years (17 users), 25–34 years (7 users), 35–44 years (5 users) and 55–64 years (1 participant).

Procedure

The study was incorporated into the Plymouth University Undergraduate Computer Science Module PRCS205/ PRDC 202—integrating project. This module aims at letting students work as groups in the completion and submission of one project as the final coursework for the module. In total, there were 10 groups; five groups were provided with the dEv framework as a guide/support resource. All students (novice developers) have the same project goals and specifications; however, each group should create and use their own development process to complete the task. Each project was provided to deliver two applications: a Web-based and a Desktop-based solution. The software evaluation is considered part of the module process, which allows groups to evaluate their own project. Throughout the evaluation process, the novice developers held a usability session with module lectures and a second session with 'student users'. After submission, the study implemented an independent usability evaluation session. A total of ten Web applications and ten Desktop applications were evaluated in the lab. The dEv group has 50% of the total applications number whilst the remaining 50% were assigned to the non-dEv group. Users testing were randomly allocated to the evaluation of an application. Throughout the duration of the evaluation lab, the users testing were presented with a series of tasks to complete using the application. All users were asked to complete the SUS survey during or following the performance of tasks. The SPSS statistical analysis software tool was used to produce the study analysis results. The study results are presented using descriptive, ANOVA and regression analysis to measure the study hypothesis.

Table 10-2: The descriptive statistics of all groups in the study (organised by groups A,B,C,D,E,F,G,H,I,J)

Group	Gender	Age	experience	occupation
A	M = 28 F = 2	1 = 20 2 = 10 3 = 0 4 = 0 5 = 0	0 = 4 ≤1 = 8 >1 = 18	UG = 16 PG = 14 FTE = 0 FTO = 0 Oth = 0
B	M = 25 F = 5	1 = 13 2 = 11 3 = 2 4 = 0 5 = 4	0 = 8 ≤1 = 10 >1 = 12	UG = 14 PG = 12 FTE = 0 FTO = 4 Oth = 0

Group	Gender	Age	experience	occupation
C	M = 28 F = 2	1 = 13 2 = 14 3 = 3 4 = 0 5 = 0	0 = 7 ≤1 = 4 >1 = 19	UG = 13 PG = 15 FTE = 0 FTO = 2 Oth = 0
D	M = 26 F = 4	1 = 16 2 = 10 3 = 4 4 = 0 5 = 0	0 = 10 ≤1 = 8 >1 = 12	UG = 17 PG = 12 FTE = 0 FTO = 1 Oth = 0
E	M = 20 F = 10	1 = 18 2 = 10 3 = 2 4 = 0 5 = 0	0 = 14 ≤1 = 8 >1 = 8	UG = 14 PG = 12 FTE = 0 FTO = 2 Oth = 2
F	M = 24 F = 6	1 = 18 2 = 10 3 = 2 4 = 0 5 = 0	0 = 10 ≤1 = 10 >1 = 10	UG = 20 PG = 8 FTE = 0 FTO = 2 Oth = 0
G	M = 27 F = 3	1 = 15 2 = 12 3 = 1 4 = 0 5 = 2	0 = 4 ≤1 = 11 >1 = 15	UG = 18 PG = 12 FTE = 0 FTO = 0 Oth = 0
H	M = 23 F = 7	1 = 22 2 = 8 3 = 0 4 = 0 5 = 0	0 = 6 ≤1 = 6 >1 = 18	UG = 19 PG = 11 FTE = 0 FTO = 0 Oth = 0
I	M = 26 F = 4	1 = 22 2 = 6 3 = 2 4 = 0 5 = 0	0 = 10 ≤1 = 6 >1 = 14	UG = 22 PG = 8 FTE = 0 FTO = 0 Oth = 0
J	M = 24 F = 6	1 = 17 2 = 7	0 = 7 ≤1 = 6	UG = 14 PG = 12

Group	Gender	Age	experience	occupation
		3 = 5 4 = 0 5 = 1	>1 = 17	FTE = 0 FTO = 2 Oth = 2
The values represent the descriptive values of users for each group testing. M represent Male, F represent Female {Gender: M (Male), F (Female)} Age: 1 (18–24), 2 (25–34), 3 (35–44), 4 (45–54), 5 (55–65), 6(56+) Programing experience: 0 (no experience), ≤1 (year or less), >1 (more than one year) Occupation: UG (Undergraduate), PG (Postgraduate), FTE (Full-time employee in education), FTO (Full-time employee in other are), Oth (Other occupation)				

10.3 Study Findings

The study results have been produced based on the novice developers and user satisfaction score with applications. This section has been divided into three parts of results, as following:

- Descriptive results for all applications
- Regression results and
- USU results.

Descriptive

The results have been divided into two groups, namely:

1. Web applications results
2. Desktop application results.

Each group's application will be individually presented. Each group's results will start with descriptive analysis, and then will present the mean values of the users' satisfaction score. The gender, age, programing experience and occupation are factors that could impact the user satisfaction score. Thus, a mean percentage score, along with the overall SUS satisfaction score, are calculated for each application. The ANOVA analysis test is presented to determine which of the four factors has a significant effect on the user satisfaction score. The study results will be divided into two groups: the first group will refer to the dEv applications containing five Web applications and five Desktop applications; the second group will be assigned all non-dEv applications containing five Web applications and five Desktop applications. Table 10-3 shows the descriptive statistics of all variables of the study, organised based on groups (A,B,C,D,E,F,G,H,I,J); these results were adopted on SUS score. (See appendix 1: D.2).

Table 10-3: The descriptive statistics of all variables of the study, organised based on groups (A,B,C,D,E,F,G,H,I,J) crosstab (on SUS)—this number is the mean

Groups		Overall %	Gender %	Age %	Experience %	Occupation %
A	Web	82.5	M = 81.6 F = 95.0 **IS**	1 = 84.5 2 = 78.5 3 = 0 4 = 0 5 = 0 **IS**	0 = 77.5 ≤1 = 80.0 >1 = 84.7 **IS**	UG = 84.3 PG = 80.3 FTE = 0 FTO = 0 Oth = 0 **IS**
	Desktop	77.6	M = 76.0 F = 100 **IS**	1 = 76.0 2 = 80.0 3 = 0 4 = 0 5 = 0 **IS**	0 = 82.5 ≤1 = 63.1 >1 =83.0 **IS**	UG = 71.8 PG = 84.2 **IS**
B	Web	85.1	M =84.2 F = 91.2 **IS**	1 = 82.1 2 = 93.0 3 = 100 4 = 0 5 = 68.7 **IS**	0 = 88.1 ≤1 = 87.5 >1 = 81.2 **IS**	UG = 84.2 PG = 91.2 FTE = 0 FTO =70.0 Oth = 0 **IS**
	Desktop	70.6	M = 71.2 F = 68.3 **IS**	1 = 67.0 2 = 87.5 3 = 70.0 4 = 0 5 = 31.2 **S**	0 = 68.1 ≤1 = 62.5 >1 = 79.1 **IS**	UG = 71.0 PG = 79.1 FTE = 0 FTO = 43.7 Oth= 0 **IS**
C	Web	90.3	M = 90.0 F = 95.0 **IS**	1 = 88.0 2 = 91.2 3 = 92.5 4 = 0 5 = 0 **IS**	0 = 91.8 ≤1 = 86.2 >1 = 90.5 **IS**	UG = 88.0 PG = 91.1 FTE = 0 FTO = 95.0 Oth = 0 **IS**
	Desktop	81.8	M = 81.0 F = 92.5	1 = 75.3 2 = 89.5	0 = 86.6 ≤1 = 75.5	UG = 79.3 PG = 83.3

Groups		Overall %	Gender %	Age %	Experience %	Occupation %
			IS	3 = 87.5 4 = 0 5 = 0 **IS**	>1 = 0 **IS**	FTE = 0 FTO = 92.5 Oth= 0 **IS**
D	Web	67.1	M = 66.7 F = 70.0 **IS**	1 = 66.1 2 = 68.7 3 = 68.7 4 = 0 5 = 0 **IS**	0 = 60.0 ≤1 = 72.0 >1 = 67.9 **IS**	UG = 68.3 PG = 65.4 FTE = 0 FTO = 0 Oth = 0 **IS**
	Desktop	82.8	M = 83.0 F = 81.2 **IS**	1 = 80.3 2 = 82.9 3 = 91.2 4 = 0 5 = 0 **IS**	0 = 78.7 ≤1 = 83.3 >1 = 86.6 **IS**	UG = 82.8 PG = 82.0 FTE =0 FTO= 87.5 Oth = 0 **IS**
E	Web	50.1	M = 54.7 F = 41.0 **IS**	1 = 49.7 2 = 51.0 3 = 50.0 4 = 0 5 = 0 **IS**	0 = 46.4 ≤1 = 59.3 >1 = 47.5 **IS**	UG = 49.6 PG = 51.2 FTE = 0 FTO = 52.5 Oth = 45.0 **IS**
	Desktop	53.3	M = 50.0 F = 60.0 **IS**	1 = 43.6 2 = 66.5 3 = 75.0 4 = 0 5 = 0 **IS**	0 = 67.5 ≤1 = 56.2 >1 = 25.6 **S**	UG = 39.2 PG = 67.9 FTE = 0 FTO = 65.0 Oth = 52.0 **IS**
F	Web	46.5	M = 44.3 F = 55.0 **IS**	1 = 37.0 2 = 38.0 3 = 92.0 4 = 0 5 = 0 **IS**	0 = 37.0 ≤1 = 45.5 >1 = 57.0 **IS**	UG = 47.5 PG = 45.5 FTE = 0 FTO= 42.5 Oth = 0 **IS**
	Desktop	68.3	M = 73.5 F = 47.5 **IS**	1 = 64.1 2 = 70.5 3 = 95.0 4 = 0 5 = 0	0 = 66.0 ≤1 = 58.0 >1 = 81.0 **IS**	UG = 60.7 PG = 88.7 FTE = 0 FTO = 62.5 Oth = 0

Groups		Overall %	Gender %	Age %	Experience %	Occupation %
				IS		**IS**
G	Web	67.8	M = 68.6 F = 62.5 **IS**	1 = 70.6 2 = 60.4 3 = 0 4 = 0 5 = 90.0 **IS**	0 = 35.0 ≤1 = 84.0 >1 = 65.9 **S**	UG = 78.8 PG = 51.2 FTE = 0 FTO= 0 Oth = 0 **S**
	Desktop	73.6	M = 73.7 F = 72.5 **IS**	1 = 80.3 2 = 72.5 3 = 92.5 4 = 0 5 = 15.0 **S**	0 = 73.7 ≤1 = 70.4 >1 = 76.4 **IS**	UG = 72.7 PG = 75.8 FTE = 0 FTO= 0 Oth = 0 **IS**
H	Web	37.0	M = 35.0 F = 42.5 **IS**	1 = 34.7 2 = 43.1 3 = 0 4 = 0 5 = 0 **IS**	0 = 30.0 ≤1 = 20.0 >1 = 45. 0 **IS**	UG = 31.9 PG = 44.5 FTE = 0 FTO = 0 Oth = 0 **IS**
	Desktop	75.5	M = 78.5 F = 63.3 **IS**	1 = 72.7 2 = 83.1 3 = 0 4 = 0 5 = 0 **IS**	0 = 69.1 ≤1 = 71.6 >1 = 78.8 **IS**	UG = 74.7 PG = 77.0 FTE = 0 FTO = 0 Oth = 0 **IS**
I	Web	70.3	M = 67.8 F = 86.2 **IS**	1 = 71.1 2 = 57.5 3 = 100 4 = 0 5 = 0 **IS**	0 = 64.0 ≤1 = 86.6 >1 = 67.8 **IS**	UG = 71.1 PG = 68.1 FTE = 0 FTO = 0 Oth = 0 **IS**
	Desktop	68.1	M = 71.7 F = 45.0 **IS**	1 = 62.9 2 = 77.5 3 = 97.5 4 = 0 5 = 0 **IS**	0 = 56.5 ≤1 = 76.6 >1 = 72.8 **IS**	UG = 62.9 PG = 82.5 FTE = 0 FTO = 0 Oth = 0 **IS**
J	Web	69.1	M = 68.3 F = 72.5	1 = 76.8 2 = 73.1	0 = 79.3 ≤1 = 50.8	UG = 72.5 PG = 64.5

Groups		Overall %	Gender %	Age %	Experience %	Occupation %
			IS	3 = 61.2 4 = 0 5 = 7.5 **IS**	>1 = 70.9 **IS**	FTE = 0 FTO = 82.5 Oth = 60.0 **IS**
	Desktop	67.1	M = 68.5 F = 61.6 **IS**	1 = 65.8 2 = 66.6 3 = 71.6 4 = 0 5 = 0 **IS**	0 = 61.6 ≤1 = 55.8 >1 = 72.7 **IS**	UG = 71.4 PG = 63.7 FTE = 0 FTO = 72.5 Oth = 52.5 **IS**

The values represent the mean values of variables.

M represent Male, F represent Female {Gender: M (Male), F (Female)}

Age: 1 (18–24), 2 (25–34), 3 (35–44), 4 (45–54), 5 (55–65), 6 (56+)

Programing experience: 0 (no experience), ≤1 (year or less), >1 (more than one years)

Occupation: UG (Undergraduate), PG (Postgraduate), FTE (Full-time employee in education), FTO (Full-time employee in other are), Oth (Other occupation)

S represent significant differences between the individuals of the groups in terms of the variable examined according to ANOVA Statistical test

IS represent insignificant

10.3.1 The dEv Groups

These applications were created by five groups, each of which has Web and Desktop applications. These five groups were asked to use the dEv framework and resources for the software development process as supporting resources.

Group A:

The results show that the overall user satisfaction with Web application was 82.5 %. **Table 10-3** shows that female users were more satisfied with the Web application than male users, giving a user satisfaction score of 95% vs 81.6%, respectively. The results show that users with no programing experience were less satisfied with the Web application, giving a score of 77.5%. Users with programming experience of a year or less gave a satisfaction score of 80.0% and those with more than one year's experience afforded a score of 84.7%. The younger (18–24 years) age group of testing users gave a score of 84.5% whilst the older (25–34 years) age group were less satisfied and accordingly gave a satisfaction score of 78.5%. Undergraduate students gave a user satisfaction score of 84.3%; however, postgraduate were less satisfied, giving a score of 80.3%.

The Desktop application was evaluated fifteen times, with the users awarding an overall user satisfaction score of 77.6%. Females were completely satisfied with the Desktop application, whilst male users, on the other hand, awarded a satisfaction score of 76.0%. The older (25–34 years) age group awarded a score of 80.0% vs the younger group (18–24 years) with a score of 76.5%. Postgraduate users were 84.2% satisfied with the application whilst the undergraduate users scored it 71.8%. Users with no programing experience were 82.5% satisfied, whereas those with one year or less experience scored the application at 63.1%. Those with more than one year's experience gave a score of 83.0% (see **Table 10-2**). Based on the one-way ANOVA test, no significant effect from a single factor was witnessed in regard to user satisfaction for both applications.

Group B:

Table 10-3 shows that the overall satisfaction score with Web application was 85.1%, with female users awarding higher satisfaction scores than male users at 91.2% vs 84.2%, respectively. Older users afforded higher user satisfaction scores, except for the age group of 55–64 years, who gave a significant satisfaction score. On the other hand, users with little or no programming experience gave higher user satisfaction scores than experienced programmers. Those with a year or less programming experience gave a score of 87.5%. Those with more than one year programming experience gave a score of 81.2%. On average, undergraduate users awarded a user satisfaction score of 84.0%, whereas postgraduates gave a score of 91.2% and, lowest of all, full-time employees gave 70.0%.

The Desktop application was awarded an overall average user satisfaction score of 70.6%. Male users awarded higher satisfaction scores than female users (71.2% vs 68.3%, respectively). Postgraduate users assigned the highest user satisfaction score. The postgraduates gave a user satisfaction score of 79.1%, undergraduates gave a score of 71.0% and full-time employees gave 43.7%. The age group of 25–34 years rated user satisfaction score at 87.5%—the highest score of any age group. Users with no programming experience awarded a user satisfaction score of 68.1%, whilst those who had a year or less gave 62.5% whilst those with more than one year's experience awarded the highest score of all at 79.1%. A one-way ANOVA test was carried out to establish which of these four factors was most influential in regard to user' satisfaction: based on the one-way ANOVA test, only the age factor in Desktop applications showed a significant impact on user satisfaction ($F(3, 11) = 5.583$, $p = .014$). To perform the post-hoc test to identify the significate between groups, the age group 55–65 years was used, containing only one participant, which prevented the test.

Thus, we combined this group with the above group. Figure 10-2 shows that there is a significant difference between the 25–34 year age group and the older than 34 years age group; however, there is no significant difference between the young group (18–24 years) all other age groups.

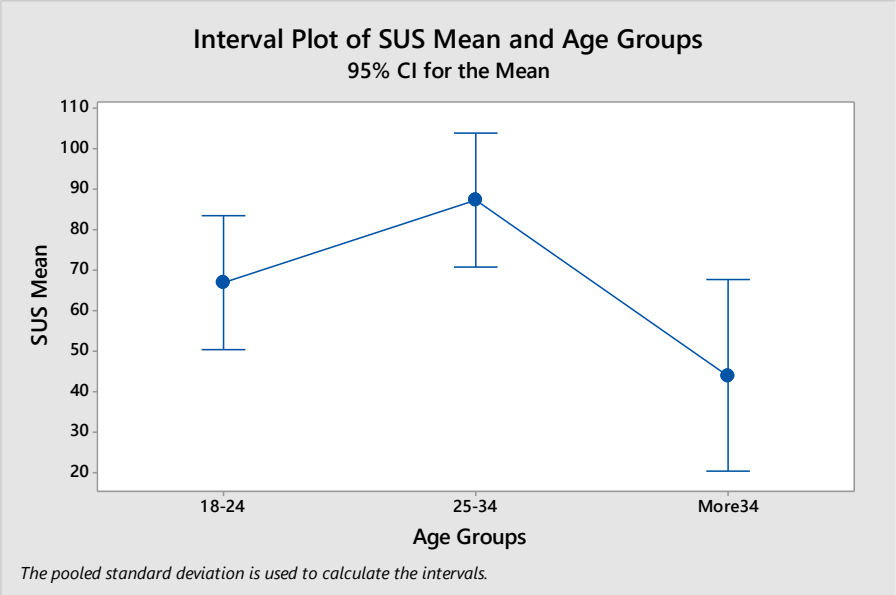


Figure 10-2: Interval plot of SUS means Vs Ages for Desktop application Group B

Group C:

The overall user satisfaction score with the Web application was 90.3%. **Table 10-3** shows that user satisfaction increased with users' age. Users in the 18–24 years age group assigned a user satisfaction score of 88.0%, whilst those in the 25–34 years age group gave a score of 91.2% and those in the 35–44 years group afforded 92.5%. On average, undergraduate users awarded a user satisfaction score of 88.0%, postgraduates gave a score of 91.1% and full-time employees gave 95.0%. The average user satisfaction was rated at 95.0% by female users and at 90.0% by male users. Users with one year programming experience gave notably lower user satisfaction scores (86.2%) when compared with those with more than one year's experience (90.5%). Users with no programming experience gave the highest average user satisfaction score at 91.8%.

The Desktop application was given an average user satisfaction score of 81.8%, with females rated their satisfaction with the application greater than male users. In term of users' occupations, undergraduates awarded a user satisfaction score of 79.3%, postgraduates awarded a score of 83.3% and full-time employees awarded 92.5%. Users with no programming experience awarded a higher average user satisfaction score than those with

one year's experience (86.6% vs 75.0%). The users in the older age group (35–44 years) gave a user satisfaction score of 87.5%, the middle age group (25–34 years) gave a score of 89.5% and the youngest group (18–24 years) gave 75.3%. Based on the one-way ANOVA test, no significant effect was witnessed as a result of any single factor on user satisfaction score across both applications.

Group D:

Users were given an overall user satisfaction score of 67.1% for the Web application, with female users assigning higher user satisfaction scores than male users. Undergraduate users also gave higher user satisfaction scores than postgraduate users. The youngest (18–24 years) age group rated the Web application with user satisfaction amounting to 66.1%, whilst both the 25–34 years and 35–44 years age groups gave a score of 68.7%. Users with no programming experience awarded a user satisfaction score of 60.0%, whilst those with more than one year's experience gave a score of 67.9% and those with a year or less gave 72.0%.

Table 10-3 shows that the Desktop application was awarded an average user satisfaction score of 82.8%. Male users awarded higher satisfaction scores than female users at 83.0% vs 81.2%, respectively. The results show that user satisfaction scores increased by both age and experience factors. Users in the 18–24 years age group gave a user satisfaction score of 80.3%, whilst users in the 25–34 years age group gave an additional 2.6% whilst those in the 35–44 years age group gave an extra 10.9% user satisfaction. Users with no programming experience gave a user satisfaction score of 78.7%, whereas those with a year or less experience gave a score of 83.3% and those with more than one year gave 86.6%. Undergraduates awarded an average user satisfaction score of 82.8%, postgraduates awarded a score of 82.0% and full-time employees awarded 87.5%. Based on a one-way ANOVA test, no significant effect could be identified as stemming from any single factor on the user satisfaction score across both applications.

Group E:

Web application evaluators gave an overall user satisfaction score of 50.1%, with male users awarding a higher satisfaction score than female users at 54.7% vs 41.0%, respectively. The results show that users with no programming experience awarded a user satisfaction score of 46.4%, with those who had a year or less awarding a score of 59.3% and those with more than one year's experience awarding 47.5%. There were three age groups involved: participant in the 18–24 years age group gave a user satisfaction score of 49.7%, users in the

25–34 years age group gave 51.0% and users in the 35–44 years age group gave 50.0%. Undergraduate users awarded a user satisfaction score of 49.6% whilst postgraduate users increased the score to 51.2%. However, full-time employees awarded a user satisfaction score of 52.5%, with the Other occupation category awarding a score of 45.0% (see Table 10-3).

The Desktop application was evaluated 15 times, with users awarding an overall user satisfaction score of 53.3%. The average user satisfaction score was rated at 60.0% by female users, with 50.0% by male users. The results show that a participant's age could positively impact user satisfaction score. The youngest (18–24 years) age group awarded a user satisfaction score of 43.6%, whilst the middle (25–34 years) age group awarded a score of 66.5% and the older (35–44 years) age group awarded 75.0%. Undergraduate users provided an overall user satisfaction score of 39.2% vs postgraduates at 67.9%, respectively, with full-time employees awarding a score of 65.0% whilst the Other occupations awarded 52.0%. The Desktop application's user satisfaction score was impacted by the level of programing experience of the users. Therefore, users with no programming experience gave a user satisfaction score of 67.5%, those with a year or less experience gave a score of 56.2% and those with more than a year's experience gave 25.6% (see Table 10-3). A one-way ANOVA test was conducted in an effort to establish which of the four factors had the most significant effect on user satisfaction. Based on the one-way ANOVA test, only programming experience in the case of Desktop application was found to show a significant impact on user satisfaction score ($F(2, 12) = 8.362$, $p = .005$). Post-hoc comparisons using the Tukey HSD test indicated that the mean score for more than 3 years' experience condition ($M = 25.6$, $SD = 15.3$) was significantly different when compared with the no experience condition ($M = 67.5$, $SD = 18.4$). However, the year or less experience condition ($M = 56.2$, $SD = 12.6$) did not significantly differ from the more than one year and no experience conditions (see Figure 10-3).

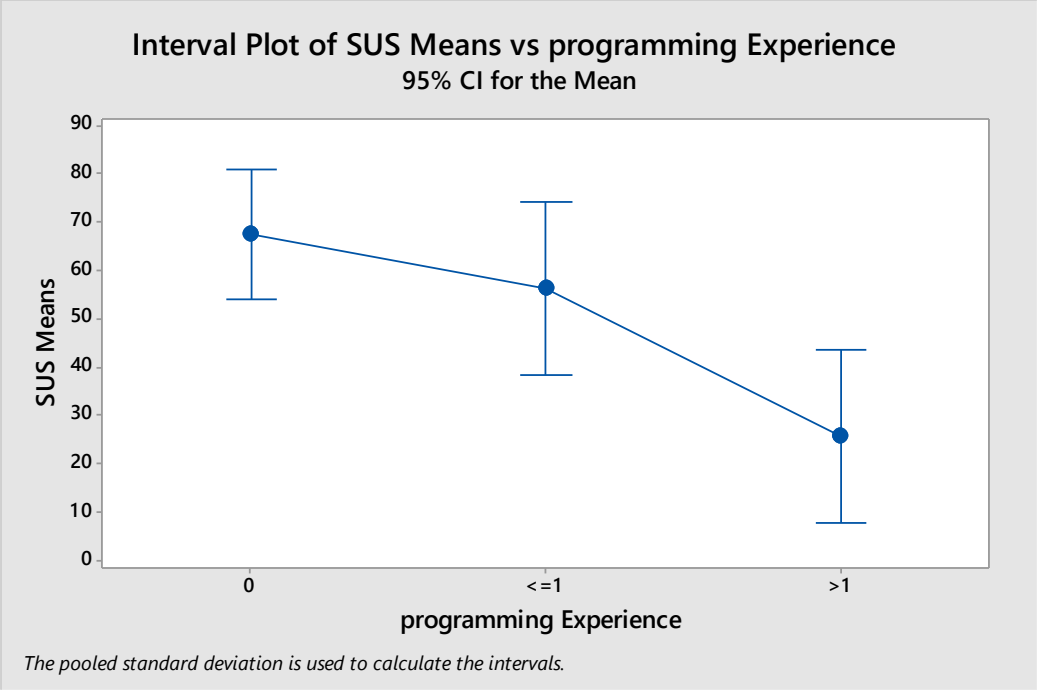


Figure 10-3: Interval plot of SUS means Vs programming experience for Desktop application Group E

10.3.2 Non-dEv Groups

These applications were created by five groups, each of which had Web and Desktop applications. These five groups were asked to use any framework and resource for the software development process.

Group F:

The Web application earned an overall user satisfaction score of 46.5%. Female users awarded higher satisfaction scores than male users at 55.0% vs 44.3%, respectively. **Table 10-3**shows that user satisfaction was increased by the level of programing experience. Thus, users that with no experience gave a user satisfaction score of 37.0%, where those with had a year or less gave a score of 45.5% and those with more than one year's experience awarded a score of 57.0%. Undergraduate users awarded a user satisfaction score of 47.5%, whilst postgraduates assigned a score of 45.5% and full-time employees awarded 42.5%. There were three age groups involved, with each group affording a different satisfaction score: the 18–24 years age group gave an average user satisfaction score of 46.1%, whilst the 25–35 years age group gave a score of 38.0% and the 35–44 years age group gave 92.5%.

The Desktop application's user satisfaction score was rated at 68.3%. Male users gave a user satisfaction score of 73.55%; however, female users gave less at 47.5%. The results show that the age groups could impact user satisfaction scores. The older (35–44 years) age group provided an average user satisfaction score of 95.0%, whilst the middle (25–35 years) age group gave a score of 70.5% and the youngest (18–24 years) age group gave 64.1%. Users with more than one year of programing experience awarded a user a satisfaction score of 81.0%, whilst those with a year or less awarded a score of 58.0% and those who with no experience awarded 66.0%. Postgraduate users gave a user satisfaction score of 88.7%, with full-time employees giving a score of 62.5% and undergraduate users affording 60.7%. Based on a one-way ANOVA test, there is no significant effect from any single factor on user satisfaction score in both applications.

Group G:

The overall satisfaction score with Web application was 67.8% .The average user satisfaction was rated at 68.6% by male users and 62.5% by female users. Undergraduate users gave a user satisfaction score higher than postgraduate users at 78.8% vs 51.2%. Three age groups were involved: the 18–24 years age group awarded a user satisfaction score of 70.6%, with the 25–34 years age group awarding a score of 60.4% and the 55–64 years age group awarding 90.0%. The results show that users with a year or less experience gave a user satisfaction score of 84.0%, whilst those with more than one year's experience gave a score of 65.9%. However, users with no experience awarded an average user satisfaction score of 35.0% (see Table 10-3). A one-way ANOVA test was conducted to establish which of these four factors had the most notable impact on user satisfaction. Based on the one-way ANOVA test, the programming experience factor showed a significant impact on user satisfaction score ($F(2, 12) = 9.548$, $p = 0.003$). Post-hoc comparisons using the Tukey HSD test indicated that the mean score for no experience condition ($M = 35.0$, $SD = 14.1$) was significantly different than one or less experience condition ($M = 84.0$, $SD = 5.7$) and more than one year condition ($M = 65.9$, $SD = 16.3$). However, the year or less experience condition did not significantly differ from the more than one year conditions (see Figure 10-4).

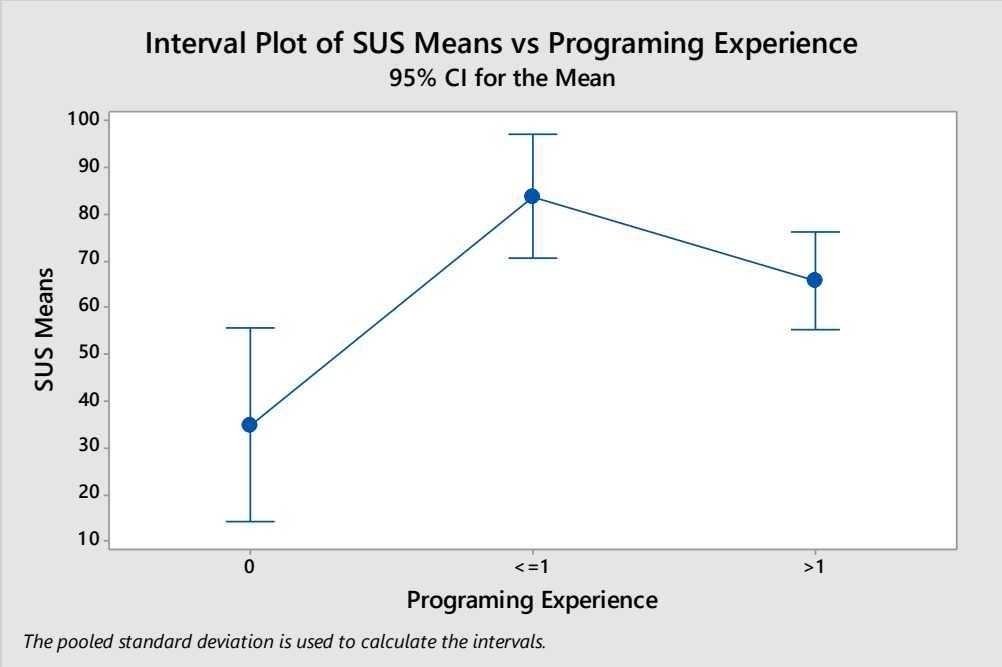


Figure 10-4: Interval plot of SUS means Vs programming experience for Web application Group G

The occupation factor showed an impact on user satisfaction score (F (1, 13) = 12.178, p = .004). Post-hoc comparisons used the Tukey HSD test, however, because the occupation contained only two types of occupation, meaning the post-hoc cannot be performed. Thus, Figure 10-5 shows that the undergraduate group has a higher mean (M =78.8, SD =11.1) than the postgraduate group (M =51.2 , SD =19.7).

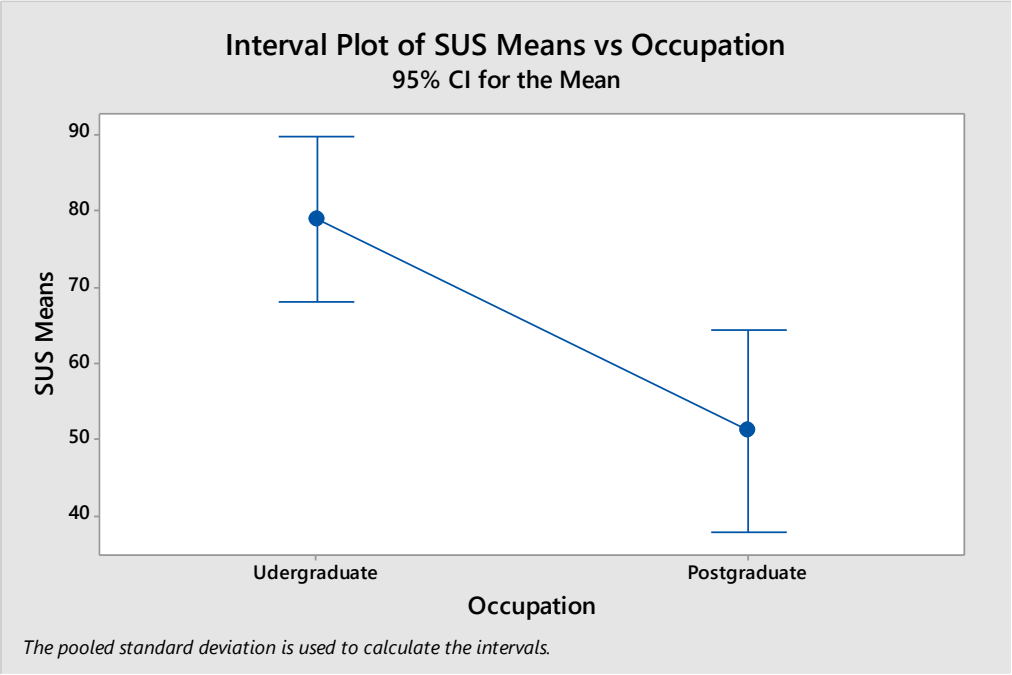


Figure 10-5: Interval plot of SUS means Vs Occupations for web application Group G

The Desktop application was evaluated 15 times, with an overall user satisfaction assigned of 73.6%. Male users gave higher satisfaction scores than female users by 1.2%, and postgraduates scored higher than undergraduates by 3.6%. Table 10-3 shows that the older (55–64 years) age group awarded the lowest user satisfaction score of 15.0%, with the 25–34 years age group awarding a score 72.5%, whilst the youngest (18–24 years) age group awarded 80.3%. Users in the 35–44 years age group awarded the highest user satisfaction score of 92.2%. Users with a year or less programing experience gave the lowest score of user satisfaction at 70.4%, with those with no experience assigning a score of 73.7 and those with more than a year giving a score of 76.4%. Based on the one-way ANOVA test, the age factor was found to have significant impact on user satisfaction score ($F(3, 11) = 4.007$, $p = .037$).Figure 10-6

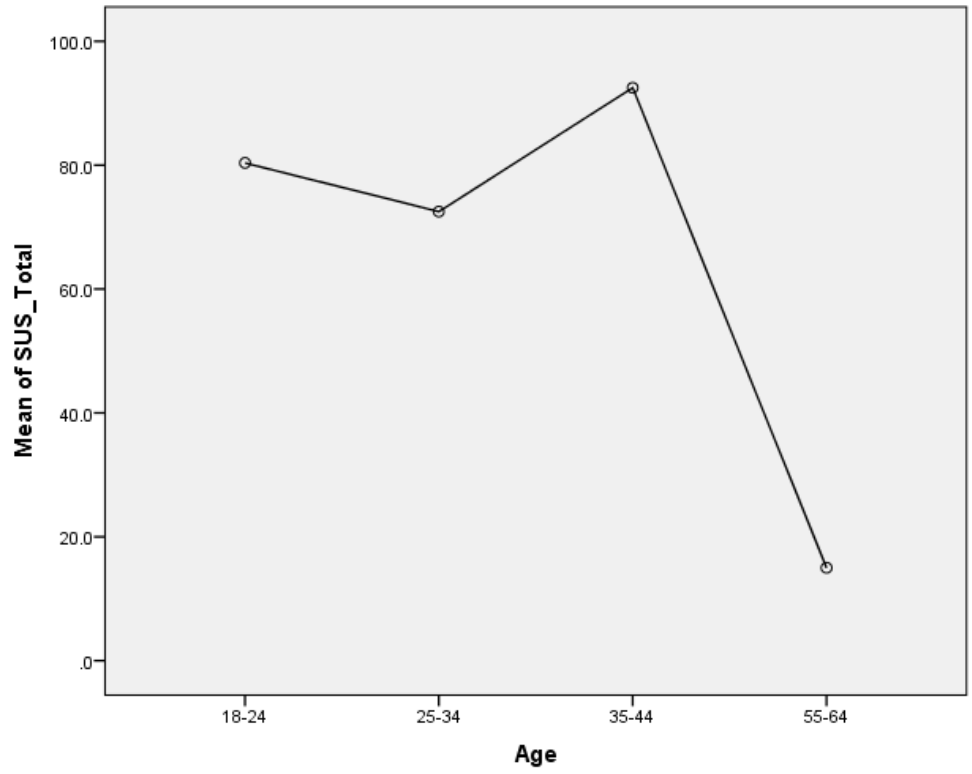


Figure 10-6: Interval plot of SUS means Vs Age for Desktop application Group G

Group H:

Table 10-3shows that THE Web application was given an overall user satisfaction score of 37.0%. Females rated the application higher than male users at 42.5% vs 35.0%, respectively,

and postgraduates awarded higher scores than undergraduates at 44.5% vs 31.9%, respectively. Users in the 25–34 years age group gave a user satisfaction score of 43.1%, and those in the 18–25 years age group gave a score of 34.7%. Users with no experience provided an overall user satisfaction score of 30.0%, whilst those with a year or less experience gave a score of 20.0% and those with more than a year's experience afforded 45.0%.

The Desktop application was given an average user satisfaction score of 75.5%. Males were awarded a user satisfaction score of 78.5%, whilst females awarded a score of vs 63.3%. Postgraduate users also gave a higher user satisfaction score than undergraduate users at 77.0% vs 74.7%, respectively. Table 10-3 shows that satisfaction score increased in line with the level of programing experience. Thus, users with no programing experience gave a user satisfaction score of 69.1%, and those who had a year or less gave a score of 71.6% whilst those with more than a year's experience gave 78.8%. The older users (25–34 years) age group gave a satisfaction score of higher than the youngest (18–24 years) age group at 83.1% vs 72.7%, respectively. Based on a one-way ANOVA test, there is no significant effect from any single factor on user satisfaction score in both applications.

Group I:

The Web application was given an overall user satisfaction score of 70.3%. Female users gave a higher satisfaction score than male users by 18.4%, and undergraduates gave a higher score than postgraduates by 3.0%. The older users (35–44 years) gave a user satisfaction score of 100%, whilst the middle (25–34 years) age group gave a score of 57.5% and the youngest (18–24 years) age group gave 71.1%. The result shows that users with no programing experience awarded a user satisfaction score of 64.0%, with those with a year or less awarded a score of 86.6% whilst those with more than one year's experience were awarded 67.8%.

The Desktop application was evaluated 15 times, with users providing an overall user satisfaction score of 68.1%. Male users gave a user satisfaction score of 71.7% vs female users with a score of 45.0%. Postgraduates awarded a user satisfaction score more than undergraduates at 82.5% vs 62.9%, respectively. Table 17 shows that the age factor increased user satisfaction overall. Thus, the younger (18–24 years) age group awarded a user satisfaction score of 62.9%, the middle (25–34 years) age group awarded a score of 77.5% and the older (35–44 years) group awarded 97.5%. Unexperienced users gave a satisfaction score of 56.5%, whilst those with a year or less awarded a score of 76.6% and those with more than one year's experience gave a score of 72.8%. Based on a one-way ANOVA test,

there was no significant effect identifiable as a result of any single factor on user satisfaction score across both applications.

Group J:

The results show that Web application had an overall user satisfaction score of 69.1%. **Table 10-3** shows that female users rated the application user satisfaction higher than male users by providing a user satisfaction score of 72.5% vs 68.3%, respectively. Users' age groups were found to have a link with user satisfaction score: the younger (18–24 years) age group gave a higher satisfaction score of 76.8%, the 25–34 years age group gave a score of 73.1%, whilst the 35–44 years age group gave a score of 61.2% and the older (55–64 years) age group assigned a lower score of 7.5%. Undergraduate users awarded a user satisfaction score of 72.5%, postgraduate users awarded a score of 64.5%, full-time employees awarded a score of 82.5% and those in the Other occupation category awarded 60.0%. Users with more than one year's programing experience gave user satisfaction a score of 70.9%, whilst those with a year or less gave a score of 50.8% and those who with no experience gave 79.3%.

The Desktop application was evaluated fifteen times, with the users awarding an overall satisfaction score of 67.1%. Female users gave a user satisfaction score of 61.6% whereas male users gave a score of 68.5%. The older (35–44 years) age group awarded a user satisfaction score of 71.6%, the middle (25–34 years) age group awarded a score of 66.6% and the younger (18–24 years) age group awarded 65.8%. Undergraduate users rated their satisfaction with the application higher than postgraduates at 71.4% vs 63.7% respectively. Full-time employees gave a user satisfaction score of 72.5%, whereas Other employees gave 52.5%. The results show that users with more than one year's programming experience gave a user satisfaction score of 72.7%, whereas those with no experience gave a score of 61.6%. However, users with a year or less of programming experience assigned a lower score of user satisfaction at 55.8%. Based on a one-way ANOVA test, no significant effect stemming from any single factor was witnessed in regard to user satisfaction score across both applications.

10.3.3 Regression Analysis

A regression test is carried out in mind of establishing which of the following factors had the most significant impact:

1. Gender
2. Age

3. Programming experience
4. Occupation
5. Novice developers with the dEv framework or developer without it
6. Type of application (Web or Desktop).

Table 10-4 shows a multiple linear regression that calculated to predict user satisfaction based on the users' gender, age, programing experience, occupation, framework used (dEv and non-dEv) and application type (Web or Desktop). A significant regression equation was found (F (6, 293) = 4.485, $p < .000$), with an R^2 of .065. Users' predicted user satisfaction is equal to 40.984 – 4.273 (Gender) + .164 (Age) + 3.764 (Programing Experience) + 1.434 (Occupation) + 9.694 (Framework) + 5.258 (Application type), where Gender is coded as 1 = Male, 2 = Female. Age is coded as 1 = 18–24 year old, 2= 25–34, 3 = 35–44, 4 = 45–54, 5 = 55–64, 6 = 65+. Programing experience is coded as 1 = no experience, 2 = a year or less and 3 = more than one year. Occupation is coded as 1 = undergraduate, 2 = postgraduate, 3 = full-time working in education, 4 = full-time working in other area, and 5 = Other occupation. Framework is coded as 1 = non-dEv framework, and 2 = dEv framework. Application type is coded as 1 = Web and 2 = Desktop. The regression test showed that programing experience was seen to have a significant effect at p value 0.031. Moreover, the type of application was seen to have a significant effect at p value .044. Furthermore, the regression analysis shows that the type of design framework had a notable effect on user satisfaction with a p value 0.000. Additionally, a bivariate correlation test was preformed to present which has a greater influence on the model predicted; thus, the software framework assessment ($\beta = .209$, $r_s^2 = .528$) was by far the best predictor of the user satisfaction model. The next variable leading the model was the assessment of programing experience ($\beta = .135$, $r_s^2 = .234$). The type of application assessment ($\beta = .113$, $r_s^2 = .154$) was the third variable leading the model. The final variable to lead the model was the gender assessment ($\beta = .068$, $r s^2 = .146$) Table 10-4 shows regressions analysis and Table 10-5 the correlations of the factor values predicted in the regression model.

Table 10-4: The regressions analysis testing for study users' gender, age, programing experience, occupation, design framework and application type factors

Model	Unstandardised Coefficients	t	Sig.
	B		
(Constant)	40.984***	4.156	.000
Gender	–4.273	-.984	.326
Age	.164	.090	.928
Pro_Exp	3.764**	2.163	.031
Occupation	1.434	.710	.478
Framework	9.694***	3.687	.000
App_code	5.258**	2.023	.044

$F = 4.485$ *Sig.*$= .000$ $R = .084$ *Adj.* $R^2 = .065$

The values represent the regression analysis of the six variables.

Gender represents the users' gender.

Age: Users' age groups

Pro_Exp: Programing level of users

Occupations: Users' occupation types

Framework: Type of design framework used to create the application.

App_code: Application type.

***, **,* indicate Significance at 0.01,0.05 and 0.1.

Table 10-5: The correlations of the factor values predicted in the regression model

Correlations								
		Gender	Age	Pro_Exp	Occupation	Framework	App_code	Unstandardised Predicted Value
Unstandardised Predicted Value	Pearson Correlation	–.383**	.103	.484**	.032	.727**	.393**	1
	Sig. (2-tailed)	.000	.074	.000	.580	.000	.000	
	N	300	300	300	300	300	300	300

The values represent the collections analysis values of the six variables.

Gender represents the participant's gender.

Age: users age groups

Pro_Exp: programing level of users

Occupations: users occupations types

Framework: type of design framework that used to create the application.

App_code: application type.

**. Correlation is significant at the 0.01 level (2-tailed). *. Correlation is significant at the 0.05 level (2-tailed).

10.3.4 SUS Score

The System Usability Scale (SUS) questionnaire has been used widely since it was first created by Booke in 1986. The number of researchers that have adopted this questionnaire to generate an overall SUS score of the products has been significant. For example, Sauro (2011a, 2011b) completed studies using the SUS questionnaire in mind of generating and accordingly interpreting the SUS scores, as shown in Figure 10-7 (Sauro, 2011b; 2011a). Bangor *et al*. (2008) completed their own studies to generate and interpret the SUS scores, and derived the results shown in Figure 10-8 (Bangor et al., 2008; Sauro, 2011a). The application centred on overall user satisfaction score will be converted to a System Usability Scale (SUS score). **Table 10-6** ranks the application by SUS score using the Sauro Scale and Bango Scale.

Figure 10-7: has been removed due to Copyright restrictions.

Figure 10-8: has been removed due to Copyright restrictions.

Table 10-6 Panel A shows the Web applications ordered by their SUS score. The top three places were taken by applications developed using the dEv framework. The next three places were achieved by applications developed without using the dEv framework. The last four places were shared by the two dEv framework applications, and were followed by two non-dEv framework-based applications. Based on the ranking of Bangor *et al*. (2008), there was only one application described as Excellent, which was a dEv framework application. Six applications were shown to be Good, whilst the remaining were described as Okay. There were two applications below the average score and eight application above the average score, where the average score equated to 68 (Sauro, 2011b).

The Desktop applications were also ranked based on the Sauro and Bangor *et al*. interpretation (see Table 10-6 Panel B). This table shows that the first three places were held by dEv applications, followed by two non-dEv applications and then one dEv application, followed by three non-dEv applications, whilst the last place was held by one dEv application. Only one application was rated below the average score of 68; however, the other nine applications were above the average score for the Sauro SUS interpretation. Based on the Bangor *et al*. (2008) interpretation, nine Desktop applications were described as Good and only one application was Okay. The top SUS score was 79 whilst the bottom score was 68.

Table 10-6: Representing the Web applications (Panel A) and Desktop application (panel B) ordered based on the SUS score. Ranking from the Sauro SUS score interpretation and Bangor et al. SUS score interoperation.

Panel A: Web applications

	dEv framework supported	Web overall mean %	SUS	Sauro Grade	Bangor *et al.* Grade	Acceptability	Adjective
C	Yes	90.3	81	A	B	Acceptable	Excellent
B	Yes	85.1	78	A-	C	Acceptable	Good
A	Yes	82.5	77	B+	C	Acceptable	Good
I	No	70.3	73	B-	C	Acceptable	Good
J	No	69.1	73	B-	C	Acceptable	Good
G	No	67.8	72	B-	C	Acceptable	Good
D	Yes	67.1	72	B-	C	Acceptable	Good
E	Yes	50.1	68	C	D	Marginal	Ok
F	No	46.5	66	C	D	Marginal	Ok
H	No	37.0	62	C-	D	Marginal	Ok

Panel B: Desktop applications

	dEv framework supported	Desktop overall mean %	SUS	Sauro Grade	Bangor *et al.* Grade	Acceptability	Adjective
D	Yes	82.8	79	B+	C	Acceptable	Good
C	Yes	81.8	78	B+	C	Acceptable	Good
A	Yes	77.6	75	B	C	Acceptable	Good
H	No	75.5	74	B	C	Acceptable	Good
G	No	73.6	74	B	C	Acceptable	Good
B	Yes	70.6	74	B	C	Acceptable	Good
F	No	68.3	73	B-	C	Acceptable	Good
I	No	68.1	73	B-	C	Acceptable	Good
J	No	67.1	72	B-	C	Acceptable	Good
E	Yes	53.3	68	C	D	Marginal	Ok

Table 10-7: Comparison between dEv and Non-dEv applications (Means)

	Web		**Desktop**	
	dEv	Non-dEv	dEv	Non-dEv
1	90.3%	70.3%	82.8%	75.5%
2	85.1%	69.1%	81.8%	73.6%
3	82.5%	67.8%	77.6%	68.3%
4	67.1%	46.5%	70.6%	68.1%
5	50.1%	37.0%	53.3%	76.1%
Overall	**75.1%**	**58.2%**	**73.3%**	**70.6%**

10.3.5 Additional Analysis (Novice developers Interview)

Semi-structured developer interviews were carried out to collect more information about the design process that the novice developers followed. There were 7 novice developers involved; 5 novice developers from the dEv group and 2 novice developers from the non-dEv group. Table 10-8 shows the main construct themes and the interoperation of these themes.

Table 10-8: The main construct themes and the interoperation of these themes

Main construct	**Interpretation of the themes identified**
Evaluation importance	Novice developers were in complete agreement that the subject of evaluation and its method are important knowledge areas for novice developers to learn and regularly apply to their design. It's important to direct consideration to the evaluation at the early stages of the development process. Furthermore, evaluation is also important in mind of improving the software as a whole; thus, users mentioned that 'there are two things make the application successful, one of them is having the correct data structure for website, mobile app or any piece of software. If you have not got this, nothing can work. The second thing is good HCI, because if you have not got that nobody will use it effectively'. Additionally, some novice developers have not established the importance of evaluation until they were involved in this study; one of the users mentioned, 'If I do any

Main construct	Interpretation of the themes identified
	other application, I will not skip the evaluation because it's important to the right direction and important for people to understand the system that you create whilst ensuring it is usable.' Five out of seven novice developers stated that all of the development stages are required for evaluation, with no exception for any stage being completed without evaluation. Thus, a developer stated that evaluation is important 'from the beginning because we need to focus on insuring the product is visibly correct, as well as making sure things are working'. The other two novice developers specified only one or two stages of the development process as important for evaluation.
Evaluation methods applied	Novice developers defined the number of evaluation methods used during the development process. The results show that 12 evaluation methods were applied by novice developers a total of 21 times. Table 10-9 shows the list of methods and the number of novice developers who used them. This result clearly shows that novice developers are willing to conduct the evaluation methods; thus, they performed a number of methods that are not included in the dEv framework, such as interviews and survey products.
Users' involvement	The interview users' results show that each developer involved at least 12 users during the evaluation process. Because user involvement is an essential element in the creation of products, a participant mentioned, 'I have more experience in usability, and I know why it is important for users to be involved in the development of any project because, without that, any project could fail'. The integrated project module structure included two usability evaluation sessions as a required stage for novice developers' projects during the development process. Furthermore, the module also recruits a pool of users for use by novice developers during the two usability evaluation sessions. Each developer should plan for user involvement; thus, some novice developers applied one method in one of the stages to involve users; some of them applied two or more methods in various stages of the software development process. The results

Main construct	Interpretation of the themes identified
	show dEv groups involved approximately 122 users during the development process, meaning they were willing to conduct the evaluation methods. Moreover, the dEv framework and resources could be encouraged to involve this number of users. Training resources are appropriate ways of letting novice developers know how to conduct usability evaluations and how users can be involved during the development process (Bruun & Stage, 2014; Howarth et al., 2009; Skov & Stage, 2005). There is no exact number of users required for involvement in software evaluation; the appropriate number of users has been argued. Nielsen states that 4±1 is adequate to determine usability problems and accordingly improve the product (Nielsen, 2000); however, conversely, Hwang & Salvendy (2010) state that the 10±2 rule is more effective and allows more usability problems to be identified, and also can be applied in any evaluation study (Hwang & Salvendy, 2010). Thus, there is a lack of consensus as to the exact number of users needing to be involved. This depends on the project and the aim of the evaluation (Dix, 2011).
Avoiding user involvement	The interviewees mentioned the reasons preventing them from involving a large numbers of users throughout the design process. They gave three reasons for limiting user numbers: the time-consuming process, developer mind-set and evaluation costs. The fact the process was time-consuming was the main reason given by the majority of novice developers as to why they avoided user involvements. Previous studies have identified that the number of evaluation methods are time-consuming, such as in the cases of cognitive walkthrough, thinking aloud and observations (Holzinger, 2005; Kjeldskov, Skov, Als & Høegh, 2004). Most novice developers found that user involvement beyond the two mandatory usability sessions would require paying users. This cost of evaluation prevented many novice developers from running additional usability sessions. This issue is still encountered by the software engineers in the industry, with

Main construct	Interpretation of the themes identified
	'discount usability' established in mind of solving this issue (Nielsen, 1994). Low-cost evaluation methods, such as heuristics evaluation and questionnaires, are considered solutions centred on evaluating products with fewer costs (Gutwin & Greenberg, 2000; Nielsen, 2007). Developer's mind-set was another reason as for why user involvement was avoided. Some novice developers considered themselves as both the developer and the user during the design process, meaning they did not feel the need to evaluate their product until the final submission. Furthermore, this challenge is already recognised as a cause to prevent novice developers involving users during the development process (C Ardito et al., 2014; Bak et al., 2008).
Evaluation improvement	Understanding the usability evaluation concept has been recognised as a challenge for novice developers (Bak, Nguyen, Risgaard & Stage, 2008). Thus, novice developers' evaluation knowledge and practices can be improved by using learning resources and software design framework. The dEv framework was built to improve novice developers' applied evaluation methods; however, the study novice developers requested more improvement, with suggestions centred on two interface factors, namely structure and content. Two of them mentioned that the interfaces are clear and understandable; however, it is important to be clearer in the design and structure. Furthermore, contents are required for more information and more options so as to avoid any novice developers dictating the use of one option. Two novice developers also mentioned that software evaluation topics should show more concern from universities; thus, a developer stated that undergraduate students should have more practice from the first year by allowing them to be involved in evaluation sessions, with second-year students evaluating and improving poor design.

Table 10-9: The type of evaluation methods and the number of developers using them

Method	Number of Users
Cognitive walkthrough	3
Focus group	1
Interview	2
Nielsen's Heuristics	1
Observations	1
Questionnaire	2
Rating scale	2
survey existed products	2
Talk aloud	1
Task scenario	3
User testing	4

10.4 Discussion

This study has been conducted in order to investigate the impacts of using an evaluation method throughout the development process on user satisfaction level. The results have been discussed in the previous section, with the results interpretation discussed in this section in an effort to answer the research question. The research question was 'Do the evaluation methods applied during the software development process impact user satisfaction?' The result section employed three metrics to answer the research question. Firstly, descriptive analysis was performed to determine the overall mean user satisfaction for each application. This process also aimed at describing the differences between the means of four factors, namely gender, age, programing experience and occupation. Secondly, the regression analysis was performed to determine which these four factors had the greatest impact on the applications' overall user satisfaction scores. Based on the regression test, we are able to garner the answer to the research question. Finally, the usability scores have been converted to Bangor and Sauro's SUS scale and interpreted according to their researches (Bangor *et al*., 2008; Sauro, 2011b).

Comparison between dEv and Non-dEv applications

The research question answered. We found that the evaluation methods conducted by the developers do have an impact on user satisfaction with the software usability. This answer

was produced based on the regression analysis (Table 10-4 and Table 10-5) and SUS tables (see Table 10-6). Table 10-10 presents a summary of the results, highlighting the usability differences between applications based on the dEv usability framework and those developed without the dEv framework.

Table 10-10: Representing the overall means for dEv and non-dEv applications

	Web		**Desktop**	
	dEv	Non-dEv	dEv	Non-dEv
1	90.3%	70.3%	82.8%	75.5%
2	85.1%	69.1%	81.8%	73.6%
3	82.5%	67.8%	77.6%	68.3%
4	67.1%	46.5%	70.6%	68.1%
5	50.1%	37.0%	53.3%	76.1%
Overall	**75.1%**	**58.2%**	**73.3%**	**70.6%**

Table 10-10 shows that Web applications developed using the dEv framework were consistently ranked ahead of those applications developed without it: for example, the highest user satisfaction score was 90.3% for a dEv-supported application, but 70.3% was the highest score for non-dEv applications. The lowest Web applications user satisfaction score for the dEv group was 50.1% vs 37.0% for the non-dEv group, respectively, with a 13.1% difference between the last Web applications in both groups. The overall user satisfaction with Web applications created based on the dEv framework was higher than non-dEv applications at 75.1% vs 58.2%, respectively. On the other hand, Desktop applications created based on the dEv framework were assigned a higher user satisfaction score than non-dEv applications, with the exception of the last application. However, overall user satisfaction with dEv Desktop application was higher than in the case of non-dEv applications at 73.3% vs 70.6%, respectively. The regression analysis results (Table 10-4 and Table 10-5) showed the dEv framework positively impacted user satisfaction. The aim of the dEv framework was centred on promoting usability evaluation methods amongst software novice developers. The results show the effectiveness of the framework in a practical study. Table 10-6 shows 9 Desktop applications were described as being of acceptable quality whilst the remaining 1 application

was described as marginal. Seven of the 10 Web applications were rated as being of acceptable quality whilst the remaining 3 applications were seen to be of marginal quality.

The dEv framework was built as a software design tool and learning resource to promote novice developers in learning and applying software evaluation methods (Almansour & Stuart, 2014). The study results show that novice developers are willing to learn and apply software evaluation methods. Number of authors believed that supporting tools and models are helpful to increase the developers ability of conducting evaluations (Aberg & Shahmehri, 2000; Lutz, Boucher, & Roustant, 2013). The dEv methods have a positive impact on final user satisfaction level. This study did not attempt to measure how effectively novice developers used the software evaluation methods; rather, this study aimed at giving novice developers an optional supporting resource that taught them a number of common evaluation methods and advised on how to use them. The dEv framework encouraged novice developers to involve users throughout the development process. The research investigator offered some services for the dEv novice developers, including a pool of users and supporting experts. These services aimed at helping the novice developers with their software evaluation. Novice developers were able to access theses services upon request during the development process. These services were recruited to establish how many novice developers would request these forms of assistance as one of the study objectives. All study novice developers had two usability sessions during the development process, as required of the course module. However, there was only one group who asked for an additional evaluation session (Group C). This group's application rated as the best Web application with an SUS grade of A, and was described as Excellent. In the final interviews, the Group C developer indicated an impact on him causing him to think about user's involvement. Other developers stated in the interviews that the involvement of additional users was difficult considering various factors, including the time-consuming process, developer mind-set and the cost of evaluation (Bak et al., 2008). More reasons ware mentioned by in another study (C Ardito et al., 2014).

Group C (who asked for an additional evaluation session) produced the second top Desktop application; however, there was only a 1.0% difference between this application and the top application. The majority of the dEv applications were better than non-dEv application, even if the novice developers did not request user or expert advice sessions. This indicates that the dEv novice developers successfully applied software evaluation methods throughout the two-module usability sessions. Regression Table 10-4 and Table 10-5 showed that three factors impacted user satisfaction scores, with the highest impacting factor identified as the dEv framework. The positive B value means the dEv framework had a higher impact than the

non-dEv framework. The users' programing experience also impacted the user satisfaction score. Users preferred the Desktop application to their Web counterparts. The results show that our framework's design is one of the key important factors associated with increased user satisfaction and software usability level.

Novice developers' Agreement on Evaluation Importance

Evaluation is an essential stage of the development process. Study developers were in complete agreement concerning the importance of evaluation; this is one of the important findings of the research as this means developers are concerned with this issue even when they do not have a wealth of experience. Conducting evaluations has been determined as a key challenge that prevents developers from applying usability evaluation (Bak *et al*., 2008). Assistant tools and learning resources help developers to show more care for users and create usable software rather than thinking about functionally testing only. Thus, the majority of study novice developers agreed that evaluation methods should be engaged with each development process stage. Furthermore, the results show that 12 evaluation methods were applied by study novice developers, meaning that he novice developers were willing to conduct the evaluations after learning how to conduct them. This study has not assessed the overall efficiency of conducting these methods; however, future studies can investigate how novice developers can do so.

Provide and Avoid User Involvement

The time taken is the most significant cause preventing novice developers from involving users during the development process. Thus, we have offered dEv groups for pool of users at any time during the development process; however, there was only one group of novice developers asked for users. The integrated project has offered two usability sessions for all module novice developers to evaluate their solutions with pool of users. These two sessions allow novice developers to conduct evaluations by themselves so as to improve solutions. Thus, the dEv groups involved 122 users using different evaluation methods.

We held the view that evaluation is costly, meaning novice developers will not involve many users if such methods are not free; however, this number of involved users clearly shows that dEv groups garner benefits from involving users. The developer mindset is another issue

leading to the creation of unusable software; some novice developers consider themselves as both developers and users of the same product. Accordingly, this challenge should be solved as soon as possible through improving novice developers' knowledge surrounding evaluation methods, alongside technical skills. Furthermore, more practice centred on evaluation—especially for undergraduate students—will be easier than for experienced novice developers to change their mind-set behaviour.

Evaluation Knowledge Improvement

Overall, the study novice developers were found to be in support of evaluation as an important topic, with novice developers needing to show more support for this process. Two novice developers supported the evaluation topic, and state that they should be fully supported by universities, with this topic included as a module for undergraduate students. Furthermore, universities also need to create a suitable environment for evaluation practicing and accordingly provide various facilities for novice developers. The dEv group's novice developers were in complete agreement that the dEv framework promotes them to learn and conduct evaluations. Moreover, dEv novice developers gave a number of improvement suggestions, meaning they were happy to have this framework and learning resource.

10.5 Conclusion and Outcomes

This research was carried out in order to test the research hypothesis. The research hypothesis suggests that software evaluation methods that are integrated into the development process do not affect user satisfaction with the application. Overall, the study results show that user satisfaction with the dEv applications was higher than in the case of non-dEv applications. Moreover, the regression analysis shows that the dEv framework has a strong impact on the user satisfaction score and the usability of the software. Essentially, we have offered dEv novice developers various on-request services, such as extra evaluation sessions, a pool of paid users and expert support, for example. However, the majority of the novice developers did not request any assistance, except in the case of one developer (Group C), who produced the top Web application and the second place for Desktop application. Additional analysis showed that most novice developers believed usability evaluations to be too time-consuming. Usability evaluation methods are not difficult to apply—even for inexperienced novice developers; guidance from a learning resource is required. This encourages them to regularly

use evaluation methods throughout the development process. It is also necessary to prevent novice developers from seeing themselves as a user. The result will be positive impacts on the usability of the resulting software application.

Chapter Eleven: Conclusions

The principle aims of this chapter are to present a set of recommendations to promote the using of usability evaluation methods during the development process, and also to explain how this research contributes to the suability perspective. This chapter also presents some of the limitations of this study and suggestions for further research.

11 Conclusions

11.1 Overview and Findings of the Research

The aim of the research was centred on creating a framework centred on encouraging novice developers to implement evaluations throughout the design process. This model's design and development were centred on improving the overall awareness of novice developers in regard to end evaluation. At the beginning of the research, seven objectives were detailed in mind of satisfying the research aim. The way in which these objectives were achieved and how the research questions were answered is discussed in the following sections.

This study was first initiated by providing background data relating to the fields associated with the research aims. Accordingly, these related fields, as well as what others did prior to this work, were presented. The first important subject, namely Human Computer Interaction (HCI), was detailed in these background theory chapters, notably in Chapter 2, with Chapter 3 focused on software development methodologies. The evaluation and usability method concepts also were discussed and considered as the key important subjects on the research, with these two subjects included in Chapter 4. The present practices in this field were also discussed in mind of examining how the development process can be improved (Chapter 5). One objective underpinning these chapters (chapters 2,5), as follows:

1. To complete a comprehensive literature review concerned with software development approaches, software usability, and HCI and UEMs (Usability Evaluation Methods) concepts.

In line with the literature review, the usability and its associated concepts still require improvement in various different aspects, such as in terms of the developer's awareness and knowledge, and the development process as a whole.

Chapter 6 shows the methodology that was applied in this research. This methodology chapter considered as guile line to achieve the research objectives and answer the research question. Mix methodology approach was chosen to include both quantitative and qualitative methods. Chapter 7 was carried out in mind of establishing the developer's knowledge pertaining to UEMs (Objective 2). Moreover, the present work also seeks to establish the novice developers' practices of completing usability assessments (Objective 3). Study data was collected from those individuals who have software programming experience at different levels through the application of an online questionnaire, the results of which indicate that

novice developers are aware of evaluation techniques, including interviews and questionnaires. Nonetheless, the heuristics evaluation method was familiar, particularly in the sense it is viewed as quick and inexpensive. The study has ascertained that novice developers would benefit from additional training and practices on evaluation methods. Importantly, 45% of the study subjects had never completed such an evaluation. Objective 3 was fulfilled in Chapter 10, when the required tasks were created by the control and Shneiderman groups with a maximum of one evaluation method (User Testing) applied by just one participant. This provides the clear indication that subjects do not have much interest in evaluation techniques. Moreover, in Chapter 11, notably concerned with integration projects, a usability evaluation was not provided by novice developers. Accordingly, it may be stated that all of these studies fulfilled objectives 2 and 3. The first research question, parts A and C, were answered by the results:

- What is the level of understating the evaluation methods?
- What is the relationship between developer's evaluation knowledge and the experience of software programming?

In line with satisfying the second and third objectives of the study, which were associated with establishing present developer practice and knowledge surrounding usability evaluations and their methods, there was the development of a model so as to ensure the study's main aim could be achieved. The dEv framework was devised in line with real user requirements so as to ensure the framework could be successfully applied. The model-creating process was explained in Chapter 7, with attention directed towards the satisfaction of novice developers. The process comprised three main steps, namely requirement collection, framework building and framework assessment. Designing the framework was centred on achieving the fourth objective, which was to develop a theoretical model, in line with objectives 2–3, in order to encourage the use of evaluation techniques in the development process.

In chapter 8, the dEv framework was developed and assessed for implementation in real practice so as to evaluate the new collaboration and how it could impact novice developers to create usable products (Objective 5). This objective was fulfilled through choosing three groups to create one task on an individual basis. All of the groups needed to follow a particular framework in order to carry out their respective tasks. The experiment detailed in Chapter 9 shows evidence to support the view that development models have an effect on the use of evaluation methods throughout the design process. The results of the experiment show

that those groups that used the dEv framework completed 65% of the total evaluation methods adopted in the experiment as a whole. Importantly, however, another two groups applied 35% of the total evaluation methods in the experiment See Table 9-5. Moreover, this experiment highlights the ability of design frameworks to increase and improve the knowledge of novice developers in evaluation. Accordingly, the dEv groups were better able to establish solutions as their ideas came from the dEv framework and enabled them to come up with solutions in line with user requirements and feedback. Accordingly, this provides an answer to Part B of the research question.

- How does the current developer's knowledge of evaluation methods impact their practice?

The prior study shows that the model had an impact on the views of novice developers concerning evaluation, which subsequently has an effect on solutions of tasks. The next study to have been carried out in mind of evaluating the dEv framework has an impact on user satisfaction with products created, in line with Objective 6. This study had the involvement of 10 different novice developers, all of whom completed the same task on an individual basis; 5 of the novice developers were supported by the dEv framework whereas the remainder were not. As has been shown in Chapter 8, users ranked the first three application as more satisfactory Table 10-6. Moreover, the testing of the regression analysis (Table 10-4) emphasises that the dEv framework has a high impact on user satisfaction; this means the dEv framework effects applications that are supported. This fulfils Objective 6 of the study. Furthermore, this support provides an answer to Part B of Research Question 1.

- How does the current developer's knowledge of evaluation methods impact their practice?

The lessons learnt as a result of the production process of the dEv framework were pivotal to the researcher, with a number of recommendations that offered at both personal and organisational levels so as to encourage the use of evaluation methods. The recommendations were developed in line with the literature so as to satisfy the context of every computer organisation. The section that follows discusses these recommendations, meeting Objective 7 of the study and providing answers to the second and third research questions.

- What steps should be taken by software organisations, such as universities, to promote the acceptance of evaluation methods in the software development process?
- How can the learning resources help novice developers to increase the acceptance of the chosen evaluation methods in development process?

Table 11-1 illustrates where each research question was answered and where each objective was fulfilled by correlating the questions and objectives with the research chapters and sections.

Table 11-1: The correlation between the research questions and objectives, and the research chapters and sections.

		Chapters (S: Section & X: Whole Chapter)										
		1	2	3	4	5	6	7	8	9	10	11
Q1	a							**S 7.1**				
	b									**S 9.3**	**S 10.3.4**	
	c							**S 7.1**				
Q2												**S 11.2**
Q3									**X**			**S 11.2**
Obj-1			**X**	**X**	**X**	**X**						
Obj-2								**S 7.1**		**S 9.3.1 & 9.3.2**	**S 10.3.4**	
Obj-3												
Obj-4								**S 7.2, 7.3 &7.4**				
Obj-5										**X**	**S 10.3.2**	
Obj-6												**X**
Obj-7												**S 11.2**

11.2 Research Recommendations

A number of recommendations have been devised by the researcher, which should be taken into consideration not only at an organisational level but also amongst practitioners.

For computer organisations:

1. Usability and Evaluation Awareness: The importance of understanding pertaining to evaluation and usability is clear, meaning both computer entities and universities need to be working in collaboration so as to enhance the awareness of novice developers. Such an improvement could be achieved through resources and classes. This would need to be planned in advance so as to ensure the various advantages associated with this process could be achieved.
2. Usability Evaluation Facilities: The right environment and adjustment to such is pivotal in order to encourage novice developers to implement evaluation. Computer organisations and universities should offer evaluation training courses for students and members, with such training regarded as pivotal for a number of reasons: primarily, software engineers need to ensure clear understanding on evaluation and usability, where additional training would allow them to self-develop and apply evaluation methods; secondarily, continuous usability evaluation training would be pivotal in decreases costs of evaluation through enhancing novice developers' skills in this regard; novice developers also would be working together and learning from one another without the need to have an external evaluator present; and lastly, sound usability evaluation methods will encourage novice developers to improve their confidence levels relating to their own products and the usability levels of such, which in turn will allow them to judge and make decisions concerning the usability of other products.

For practitioners:

1. Practitioners need to view evaluation and usability from a wide lens and ensure understanding of the fact that usability is not concerned with testing and coding only, with priority afforded to far more than end-of-design testing.

2. Practitioners need to be confident in the usability level of their product prior to submitting the design. Moreover, reasons behind design choice, style and approach need to be made clear and justified.
3. Practitioners should hold the view that evaluations sessions can be completed by themselves with the aim of gathering user feedback in an effort to enhance their products from the early stages of the development process. Moreover, specialist evaluators no longer will need to be involved in the design process.

11.3 The Contribution and Novelty of this Research

Despite the fact that the knowledge surrounding overall usability and software assessment approaches is lacking, it remains that, similarly, there is a lack of understanding in terms of how products can be professionally evaluated and how such assessment sessions can be carried out. In dealing with this gap in learning resources, an investigation research was carried out in order to analyse the present status of the practices and knowledge of evaluation amongst novice developers in order to create a set of suggestions aimed at improving the use of such techniques throughout the development and promotion processes of practitioners and organisations. The various points provide a summary of the key contributions made by the present work, in addition to an outline of the thesis's individuality.

- The proposition of the development model (dEv framework) combines software development and software evaluation, as detailed in . This integration positions the framework as encouraging practitioners to assess at all individual stages of the development process. This section details the fact that our development framework users created their solution in line with the evaluation methods chapter 8.
- The most critical contribution made by this thesis is the dEv learning resource, as discussed in Section (7.4). This learning tool provides a number of contributions: primarily, it enhances developer awareness concerning assessment techniques and how these can be implemented; and secondarily, the tool is able to be applied so as to enhance novice developers' abilities to make choices surrounding software usability.
- A number of recommendations have been made by this work, providing computer organisations and professionals with support and valuable evaluation methods, as well as the ability to improve the usability of the software through sound planning and fewer costs.

11.4 Research Limitations

Regardless of the fact that the study objectives have been fulfilled, it remains that the work suffered a number of limitations, which could have an effect on the findings. These limitations are discussed below:

- Despite the fact that the framework and learning resource devised in this work provides a valuable impact on the products of the developer, it remains that there is a pressing need for additional services that could be pivotal in enhancing the overall productivity of the framework.
- The organisation and sample utilised in order to evaluate the tool was applicable to the study, especially during the time at which the assessment was carried out; however, the dEv tool is able to be evaluated in various contexts, such as in computer organisations, universities, different countries, and amongst those with various levels of programming experience.
- The framework assessment period with the involvement of the undergraduate students was relatively short, meaning the very best results would require a longer period of time. Moreover, framework integration with the long-term project is essential not only for product development but also for the framework.
- The learning resource suggested, both prior to and following improvement, was not tested through the gathering of data from online tools, including Google Analytics.

11.5 Further Research

The present study has provided a starting stage for gaining understanding of, and accordingly encouraging, evaluation methods and how these can be completed by software engineers throughout the course of the design development process. Overall, the study's findings and limitations can guide subsequent study efforts in various efforts that are recognised as important for academic purposes. The following are areas for consideration:

- The dEv framework and its learning resources are pivotal in the completion and evaluation of products. Accordingly, the framework should encompass additional

services and tools geared towards assisting novice developers in garnering benefits from the learning resources.

- The suggested framework requires additional examining in order to evaluate the pros and cons of this model with organisations' products, as well as in order to gather in-depth feedback through the application of interviews and questionnaires from organisations and HCI professionals.
- Future studies also need to prioritise integrating this study's framework with first-year computer science students who require further education and training on design evaluation and product assessment. Moreover, comparison studies should be carried out in mind of drawing a contrast between the results of framework adoption alongside non-adoption.

Appendix 1 :

Appendix A.1: Design Evaluation Experience Questionnaire

This survey is being conducted as part of a PhD research degree on software design. This research will focus on design evaluation methods and specifically discount usability. The survey will investigate the user's experience of evaluating the design of software systems and usability. Since design evaluation is often underestimated, we hope that the survey will also help us to establish what is needed (software, tutorials etc.) in order to increase its use. Hopefully, these new requirements will enable us to develop a system to support discount usability. The questionnaire comprises three sections, made up of a number of questions. The sections are: 1. Background information. 2. Users' experiences of software evaluation. 3. Requirements of Discount Usability (DU) evaluation methods. Thank you for your participation. If you have any queries whatsoever, please do not hesitate to contact me: Fahad Almansour School of Computing and Mathematics Plymouth University Plymouth United KingdomPL4 8AAMail to: fahad.almansour@plymouth.ac.uk

Consent Form

Dear Participant You have been invited to participate in this design evaluation methods study, that is supported by Plymouth University. If you are interested in participating in the study, please take some time to read the following information carefully. It is important for us to ensure the study and its procedures, are clear to you before you consent to proceed. This survey is solely designed for adult participants. If you are under 18 years, PLEASE DO NOT ANSWER THIS SURVEY. Any participants who are 18 years old and over may take part in the survey. All participants have the right to withdraw at any time, before they submit their data electronically at the end of the survey. All answers will be treated confidentially and respondents will remain anonymous throughout the collection, storage and publication of the data and all subsequent research material. Responses will be collected online and stored in a secure database. Individual responses will be treated as confidential at all times and the data will be presented in such a way that your identity cannot be connected with any published data. Once the survey has been taken offline, participant responses will be extracted and then statistically analysed. Results are likely to be published in a suitable academic conference and/or journal. In addition, these results will be used and published as part of a PhD thesis. If you would like to have a summary of the results, just let me know and I will

organise this for you. If you have any questions whatsoever about the study, please do not hesitate to contact the researcher: Fahad Almansour (details below).

☐ I agree to the following: Prior to clicking the submit button at the end of the online study, I understand that I am free to withdraw my responses at any time. I understand that my anonymity is guaranteed, unless I expressly state otherwise. I understand that the Principal Investigator of this work has attempted, as far as possible, to avoid any risks, and that safety and health risks will have been separately assessed by appropriate authorities (e.g. under COSHH regulations). Thank you for your participation. If you have any queries whatsoever, please do not hesitate to contact me: Fahad Almansour School of Computing and Mathematics Plymouth University Plymouth, United Kingdom.PL4 8AAemail: fahad.almansour@plymouth.ac.uk

Section One: Background information

1. What is your gender?

○ Male

○ Female

2. Which age group do you belong to?

○ 18-24

○ 25-34

○ 35-44

○ 45-54

○ 55-64

○ 65+

3. How much programming experience do you have?

○ Less than 1 year

○ 1 year

○ 2 years

○ 3 years

○ More than 3 years

4. Please select which statement best describes your occupation

○ I am an Undergraduate student

○ I am a Postgraduate student

○ I work full time within Education

○ I work full time in another area

○ Other, please specify... _____

5. What is your country of residence?

○ United Kingdom

○ United States

○ Afghanistan

○ Albania

... other additional choices hidden ...

Section two: Users' experiences of software evaluation

1. What does the term "design evaluation" mean to you?

Before you complete the next section, please take some time to consider our working definition of "design evaluation" which is widely used within the HCI community. This definition is loosely based on AEA (American Evaluation Association) and it defines design evaluation as the systematic approach to the gathering and analysis of data to define requirements, to assess the merit, worth and significance of the item undergoing evaluation.

1. Have you ever conducted an evaluation of your own design?

○ Yes

○ No

1.1 How many times have you evaluated your design?

○ Once

○ Twice

○ Three times

○ More than three times

1.2 What type of evaluation have you used?

1.1 Is there any specific reason why you chose not?

2. Have you participated in an evaluation study before as a user?

○ Yes

○ No

3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)?

□ Thinking aloud protocol (user express thoughts)

□ HE (quality principles)

□ Task scenario (user complete given task)

□ Questionnaire (user fills in answers)

□ Observation (user behavior is noted/recorded)

□ Interview (discussion with user)

□ Focus groups (group interview)

□ Other, please specify... _____

4. Have you heard about cheap and expensive evaluation methods?

○ Yes

○ No

5. Have you ever heard about Discount Usability?

○ Yes

○ No

Third section: Requirements of design evaluation (dE) methods

To what extent do you agree with the following statements:

	Strongly Agree	Agree	Neutral	Strongly Disagree	Disagree
Students of software engineering should be encouraged to use methods of design evaluation regularly.	○	○	○	○	○
Ideally, software engineers should develop their technical skills alongside their design evaluation skills.	○	○	○	○	○
Once I learn how to run design evaluations, I will be more likely to do them.	○	○	○	○	○

	Strongly Agree	Agree	Neutral	Strongly Disagree	Disagree
I always use an external evaluator as I don't have the time to get up to speed on this topic	○	○	○	○	○

To what extent do you agree with the following statements:

	Strongly Agree	Agree	Neutral	Strongly Disagree	Disagree
Resources that teach you how to evaluate your design are important.	○	○	○	○	○
I want to learn the basics about design evaluation so I can get on and do it as soon as possible.	○	○	○	○	○
I want to invest my time in order to learn all about the topic of design evaluation.	○	○	○	○	○
It's important for every software engineer to know how to evaluate their software.	○	○	○	○	○
Teaching on design evaluation should be mandatory for all software engineering students.	○	○	○	○	○

Appendix A.2: Design Evaluation Experience Questionnaire Analysis and Results

Statistics

3. How much programming experience do you have?

N	Valid	84
	Missing	0
Mean		1.58
Sum		133

3. How much programming experience do you have?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	year and Less than 1 year	35	41.7	41.7	41.7
	more than1 year	49	58.3	58.3	100.0
	Total	84	100.0	100.0	

Experience Level Vs Discount Usability

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 5. Have you ever heard about Discount Usability?	84	100.0%	0	0.0%	84	100.0%

3. How much programming experience do you have? * 5. Have you ever heard about Discount Usability?

Crosstabulation

Count

		5. Have you ever heard about Discount Usability?		Total
		Yes	No	
3. How much programming experience do you have?	year and Less than 1 year	5	30	35
	more than1 year	11	38	49
Total		16	68	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.882[a]	1	.348		
Continuity Correction[b]	.432	1	.511		
Likelihood Ratio	.905	1	.341		
Fisher's Exact Test				.409	.258
Linear-by-Linear Association	.872	1	.350		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 6.67.

b. Computed only for a 2x2 table

Experience Level Vs Evaluation Conduct

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 1. Have you ever conducted an evaluation of your own design?	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 1. Have you ever conducted an evaluation of your own design? Crosstabulation

Count

		1. Have you ever conducted an evaluation of your own design?		Total
		Yes	No	
3. How much programming experience do you have?	year and Less than 1 year	16	19	35
	more than1 year	30	19	49
Total		46	38	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.983[a]	1	.159		
Continuity Correction[b]	1.406	1	.236		
Likelihood Ratio	1.985	1	.159		

Fisher's Exact Test				.187	.118
Linear-by-Linear Association	1.959	1	.162		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.83.

b. Computed only for a 2x2 table

Experience Level Vs Evaluation User

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 2. Have you participated in an evaluation study before as a user?	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 2. Have you participated in an evaluation study before as a user? Crosstabulation

Count

		2. Have you participated in an evaluation study before as a user?		Total
		Yes	No	
3. How much programming experience do you have?	year and Less than 1 year	18	17	35
	more than1 year	28	21	49
Total		46	38	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.269[a]	1	.604		
Continuity Correction[b]	.088	1	.767		
Likelihood Ratio	.269	1	.604		
Fisher's Exact Test				.660	.383
Linear-by-Linear Association	.266	1	.606		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.83.

b. Computed only for a 2x2 table

Experience Level Vs Thinking Aloud Method

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Thinking aloud protocol (user express thoughts)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Thinking aloud protocol (user express thoughts) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Thinking aloud protocol (user express thoughts)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	17	18	35
	more than1 year	21	28	49
Total		38	46	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.269[a]	1	.604		
Continuity Correction[b]	.088	1	.767		
Likelihood Ratio	.269	1	.604		
Fisher's Exact Test				.660	.383
Linear-by-Linear Association	.266	1	.606		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.83.

b. Computed only for a 2x2 table

Experience Level Vs HE Method

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? HE (quality principles)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? HE (quality principles) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? HE (quality principles)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	3	32	35
	more than1 year	14	35	49
Total		17	67	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	5.059[a]	1	.024		
Continuity Correction[b]	3.896	1	.048		
Likelihood Ratio	5.513	1	.019		
Fisher's Exact Test				.029	.021
Linear-by-Linear Association	4.999	1	.025		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 7.08.

b. Computed only for a 2x2 table

Experience Level Vs Task Scenario Method

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Task scenario (user complete given task)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Task scenario (user complete given task) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Task scenario (user complete given task)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	16	19	35
	more than1 year	24	25	49
Total		40	44	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.087[a]	1	.768		
Continuity Correction[b]	.005	1	.941		
Likelihood Ratio	.087	1	.768		
Fisher's Exact Test				.827	.471
Linear-by-Linear Association	.086	1	.769		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 16.67.

b. Computed only for a 2x2 table

Experience Level Vs Questionnaire Method

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Questionnaire (user fills in answers)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Questionnaire (user fills in answers) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Questionnaire (user fills in answers)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	25	10	35
	more than1 year	33	16	49
Total		58	26	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.159[a]	1	.690		
Continuity Correction[b]	.025	1	.873		
Likelihood Ratio	.160	1	.689		
Fisher's Exact Test				.812	.439
Linear-by-Linear Association	.157	1	.692		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 10.83.

b. Computed only for a 2x2 table

Experience Level Vs Observation

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Observation (user behavior is noted/recorded)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Observation (user behaviour is noted/recorded) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Observation (user behavior is noted/recorded)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	14	21	35
	more than1 year	23	26	49
Total		37	47	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.399[a]	1	.528		
Continuity Correction[b]	.167	1	.683		
Likelihood Ratio	.400	1	.527		
Fisher's Exact Test				.656	.342
Linear-by-Linear Association	.394	1	.530		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.42.

b. Computed only for a 2x2 table

Experience Level Vs Interviews

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Interview (discussion with user)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Interview (discussion with user) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Interview (discussion with user)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	15	20	35
	more than1 year	27	22	49
Total		42	42	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.224[a]	1	.268		
Continuity Correction[b]	.784	1	.376		
Likelihood Ratio	1.228	1	.268		
Fisher's Exact Test				.376	.188
Linear-by-Linear Association	1.210	1	.271		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 17.50.

b. Computed only for a 2x2 table

Experience Level Vs Focus groups

Case Processing Summary

	Cases					
	Valid		Missing		Total	
	N	Percent	N	Percent	N	Percent
3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Focus groups (group interview)	84	100.0%	0	0.0%	84	100.0 %

3. How much programming experience do you have? * 3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Focus groups (group interview) Crosstabulation

Count

		3. Which of these evaluation methods have you dealt with before (either as evaluator or participant)? Focus groups (group interview)		Total
		select	non select	
3. How much programming experience do you have?	year and Less than 1 year	11	24	35
	more than1 year	21	28	49
Total		32	52	84

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.131[a]	1	.288		
Continuity Correction[b]	.698	1	.403		
Likelihood Ratio	1.142	1	.285		
Fisher's Exact Test				.364	.202
Linear-by-Linear Association	1.117	1	.290		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 13.33.

b. Computed only for a 2x2 table

Experience Level Vs Heard about Cheap and Expensive Evaluation Methods

Crosstab

		4. Have you heard about cheap and expensive evaluation methods?		Total
		Yes	No	
3. How much programming experience do you have?	**Less than or 1 year**	10	25	35
		28.6%	71.4%	100.0%
		40.0%	42.4%	41.7%
		-.4	.4	
	more than1 year	15	34	49
		30.6%	69.4%	100.0%
		60.0%	57.6%	58.3%
		.4	-.4	
Total		25	59	84
		29.8%	70.2%	100.0%
		100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.041[a]	1	.840		
Continuity Correction[b]	.000	1	1.000		
Likelihood Ratio	.041	1	.840		
Fisher's Exact Test				1.000	.518
Linear-by-Linear Association	.040	1	.841		
N of Valid Cases	84				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 10.42.

b. Computed only for a 2x2 table

Frequencies

Statistics

		To what extent do you agree with the following statements: | Students of	To what extent do you agree with the following statements: | Ideally,	To what extent do you agree with the following statements: | Once I learn	To what extent do you agree with the following statements: | I always use an	To what extent do you agree with the following statements: | Resources that	To what extent do you agree with the following statements: | I want to learn	To what extent do you agree with the following statements: | I want to invest	To what extent do you agree with the following statements: | It's important	To what extent do you agree with the following statements: | Teaching on
N	Valid	84	84	84	84	84	84	84	84	84
	Missing	0	0	0	0	0	0	0	0	0
Mean		1.11	1.15	1.31	1.69	1.13	1.26	1.54	1.14	1.36
Median		1.00	1.00	1.00	2.00	1.00	1.00	1.00	1.00	1.00
Std. Deviation		.348	.425	.580	.744	.404	.562	.735	.385	.614
Sum		93	97	110	142	95	106	129	96	114

Statement 1:

To what extent do you agree with the following statements: | Students of software engineering should be encouraged to use methods of design evaluation regularly.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	76	90.5	90.5	90.5
	Neutral	7	8.3	8.3	98.8
	Disagree	1	1.2	1.2	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | Students of software engineering should be encouraged to use methods of design evaluation regularly.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	31	4	0	35
		88.6%	11.4%	0.0%	100.0%
		40.8%	57.1%	0.0%	41.7%
	more than1 year	45	3	1	49
		91.8%	6.1%	2.0%	100.0%
		59.2%	42.9%	100.0%	58.3%
Total		76	7	1	84
		90.5%	8.3%	1.2%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.428[a]	2	.490
Likelihood Ratio	1.779	2	.411
Linear-by-Linear Association	.025	1	.874
N of Valid Cases	84		

a. 4 cells (66.7%) have expected count less than 5. The minimum expected count is .42.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	-.017	.108	-.158	.875[c]
Ordinal by Ordinal	Spearman Correlation	-.051	.111	-.461	.646[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 2:

To what extent do you agree with the following statements: | Ideally, software engineers should develop their technical skills alongside their design evaluation skills.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	73	86.9	86.9	86.9
	Neutral	9	10.7	10.7	97.6
	Disagree	2	2.4	2.4	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | Ideally, software engineers should develop their technical skills alongside their design evaluation skills.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	30	5	0	35
		85.7%	14.3%	0.0%	100.0%
		41.1%	55.6%	0.0%	41.7%
	more than1 year	43	4	2	49
		87.8%	8.2%	4.1%	100.0%
		58.9%	44.4%	100.0%	58.3%
Total		73	9	2	84
		86.9%	10.7%	2.4%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	2.153[a]	2	.341
Likelihood Ratio	2.867	2	.238
Linear-by-Linear Association	.047	1	.828
N of Valid Cases	84		

a. 3 cells (50.0%) have expected count less than 5. The minimum expected count is .83.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.024	.102	.216	.830[c]
Ordinal by Ordinal	Spearman Correlation	-.021	.110	-.193	.848[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 3

To what extent do you agree with the following statements: | Once I learn how to run design evaluations, I will be more likely to do them.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	63	75.0	75.0	75.0
	Neutral	16	19.0	19.0	94.0
	Disagree	5	6.0	6.0	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | Once I learn how to run design evaluations, I will be more likely to do them.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	26	7	2	35
		74.3%	20.0%	5.7%	100.0%
		41.3%	43.8%	40.0%	41.7%
	more than1 year	37	9	3	49
		75.5%	18.4%	6.1%	100.0%
		58.7%	56.3%	60.0%	58.3%
Total		63	16	5	84
		75.0%	19.0%	6.0%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	.038[a]	2	.981
Likelihood Ratio	.038	2	.981
Linear-by-Linear Association	.004	1	.949
N of Valid Cases	84		

a. 2 cells (33.3%) have expected count less than 5. The minimum expected count is 2.08.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	-.007	.109	-.063	.950[c]
Ordinal by Ordinal	Spearman Correlation	-.012	.109	-.107	.915[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 4

To what extent do you agree with the following statements: | I always use an external evaluator as I don't have the time to get up to speed on this topic

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	40	47.6	47.6	47.6
	Neutral	30	35.7	35.7	83.3
	Disagree	14	16.7	16.7	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | I always use an external evaluator as I don't have the time to get up to speed on this topic			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	18	12	5	35
		51.4%	34.3%	14.3%	100.0%
		45.0%	40.0%	35.7%	41.7%
	more than1 year	22	18	9	49
		44.9%	36.7%	18.4%	100.0%
		55.0%	60.0%	64.3%	58.3%
Total		40	30	14	84
		47.6%	35.7%	16.7%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	.421[a]	2	.810
Likelihood Ratio	.423	2	.809
Linear-by-Linear Association	.415	1	.519
N of Valid Cases	84		

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.83.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.071	.108	.642	.523[c]
Ordinal by Ordinal	Spearman Correlation	.071	.108	.640	.524[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 5

To what extent do you agree with the following statements: | Resources that teach you how to evaluate your design are important.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	75	89.3	89.3	89.3
	Neutral	7	8.3	8.3	97.6
	Disagree	2	2.4	2.4	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | Resources that teach you how to evaluate your design are important.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?		33	2	0	35
	Less than or 1 year	94.3%	5.7%	0.0%	100.0%
		44.0%	28.6%	0.0%	41.7%
		42	5	2	49
	more than1 year	85.7%	10.2%	4.1%	100.0%
		56.0%	71.4%	100.0%	58.3%
		75	7	2	84
Total		89.3%	8.3%	2.4%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	2.090[a]	2	.352
Likelihood Ratio	2.839	2	.242
Linear-by-Linear Association	2.001	1	.157
N of Valid Cases	84		

a. 4 cells (66.7%) have expected count less than 5. The minimum expected count is .83.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.155	.080	1.423	.158[c]
Ordinal by Ordinal	Spearman Correlation	.140	.095	1.282	.203[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 6

To what extent do you agree with the following statements: | I want to learn the basics about design evaluation so I can get on and do it as soon as possible.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	67	79.8	79.8	79.8
	Neutral	12	14.3	14.3	94.0
	Disagree	5	6.0	6.0	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | I want to learn the basics about design evaluation so I can get on and do it as soon as possible.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	29	5	1	35
		82.9%	14.3%	2.9%	100.0%
		43.3%	41.7%	20.0%	41.7%
	more than1 year	38	7	4	49
		77.6%	14.3%	8.2%	100.0%
		56.7%	58.3%	80.0%	58.3%
Total		67	12	5	84
		79.8%	14.3%	6.0%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.038[a]	2	.595
Likelihood Ratio	1.131	2	.568
Linear-by-Linear Association	.727	1	.394
N of Valid Cases	84		

a. 2 cells (33.3%) have expected count less than 5. The minimum expected count is 2.08.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.094	.101	.851	.397[c]
Ordinal by Ordinal	Spearman Correlation	.074	.106	.672	.503[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 7

To what extent do you agree with the following statements: | I want to invest my time in order to learn all about the topic of design evaluation.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	51	60.7	60.7	60.7
	Neutral	21	25.0	25.0	85.7
	Disagree	12	14.3	14.3	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | I want to invest my time in order to learn all about the topic of design evaluation.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	23	9	3	35
		65.7%	25.7%	8.6%	100.0%
		45.1%	42.9%	25.0%	41.7%
	more than1 year	28	12	9	49
		57.1%	24.5%	18.4%	100.0%
		54.9%	57.1%	75.0%	58.3%
Total		51	21	12	84
		60.7%	25.0%	14.3%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.631[a]	2	.442
Likelihood Ratio	1.716	2	.424
Linear-by-Linear Association	1.273	1	.259
N of Valid Cases	84		

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.00.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.124	.103	1.130	.262[c]
Ordinal by Ordinal	Spearman Correlation	.110	.106	1.001	.320[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 8

To what extent do you agree with the following statements: | It's important for every software engineer to know how to evaluate their software.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	73	86.9	86.9	86.9
	Neutral	10	11.9	11.9	98.8
	Disagree	1	1.2	1.2	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | It's important for every software engineer to know how to evaluate their software.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	30	5	0	35
		85.7%	14.3%	0.0%	100.0%
		41.1%	50.0%	0.0%	41.7%
	more than1 year	43	5	1	49
		87.8%	10.2%	2.0%	100.0%
		58.9%	50.0%	100.0%	58.3%
Total		73	10	1	84
		86.9%	11.9%	1.2%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.010[a]	2	.604
Likelihood Ratio	1.369	2	.504
Linear-by-Linear Association	.000	1	1.000
N of Valid Cases	84		

a. 3 cells (50.0%) have expected count less than 5. The minimum expected count is .42.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.000	.107	.000	1.000[c]
Ordinal by Ordinal	Spearman Correlation	-.026	.110	-.231	.818[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Statement 9

To what extent do you agree with the following statements: | Teaching on design evaluation should be mandatory for all software engineering students .

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Agree	60	71.4	71.4	71.4
	Neutral	18	21.4	21.4	92.9
	Disagree	6	7.1	7.1	100.0
	Total	84	100.0	100.0	

Crosstab

		To what extent do you agree with the following statements: | Teaching on design evaluation should be mandatory for all software engineering students.			Total
		Agree	Neutral	Disagree	
3. How much programming experience do you have?	**Less than or 1 year**	26	8	1	35
		74.3%	22.9%	2.9%	100.0%
		43.3%	44.4%	16.7%	41.7%
	more than1 year	34	10	5	49
		69.4%	20.4%	10.2%	100.0%
		56.7%	55.6%	83.3%	58.3%
Total		60	18	6	84
		71.4%	21.4%	7.1%	100.0%
		100.0%	100.0%	100.0%	100.0%

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.669[a]	2	.434
Likelihood Ratio	1.859	2	.395
Linear-by-Linear Association	.812	1	.367
N of Valid Cases	84		

a. 2 cells (33.3%) have expected count less than 5. The minimum expected count is 2.50.

Symmetric Measures

		Value	Asymp. Std. Error[a]	Approx. T[b]	Approx. Sig.
Interval by Interval	Pearson's R	.099	.101	.900	.371[c]
Ordinal by Ordinal	Spearman Correlation	.072	.106	.652	.516[c]
N of Valid Cases		84			

a. Not assuming the null hypothesis.

b. Using the asymptotic standard error assuming the null hypothesis.

c. Based on normal approximation.

Appendix B.1: dEv Evaluation & Users Satisfaction Survey

1. What is your gender?

○ Male

○ Female

2. Which age group do you belong to?

○ 18-24

○ 25-34

○ 35-44

○ 45-54

○ 55-64

○ 65+

3. How many times have you used this software

○ Once

○ Twice

○ 3 times

○ More than 3 times

4. How do you rate your evaluation experience

○ Beginner

○ Intermediate

○ Expert

To what extent do you agree with the following statements:

	Strongly Agree	Agree	Neutral	Strongly Disagree	Disagree
Software has clear interface structure	○	○	○	○	○
Software has easy navigation	○	○	○	○	○
Menu that on the left side helps me find information easily	○	○	○	○	○
References that are placed on the right side of every topic are helpful	○	○	○	○	○
References encourage me to expand the topic for more information	○	○	○	○	○

To what extent do you agree with the following statements:

	Strongly	Agree	Neutral	Strongly	Disagree

	Agree				Disagree
Content structure helps me to clearly understand the presented topic	○	○	○	○	○
Using images is helpful to understand the topic	○	○	○	○	○
Using videos is helpful to understand the topic	○	○	○	○	○
The videos provided reduced learning time	○	○	○	○	○
Using links through text to jump between different topics is obvious	○	○	○	○	○

please select how likely it is that you would recommend our software to a friend or colleague?

- ○ 10
- ○ 9
- ○ 8
- ○ 7
- ○ 6
- ○ 5
- ○ 4
- ○ 3
- ○ 2
- ○ 1
- ○ 0

For each of the following categories, please rate your satisfaction level with our software:

	Very Dissatisfied	Dissatisfied	Neutral	Satisfied	Very Satisfied
Software Interface	○	○	○	○	○
Overall Appearance	○	○	○	○	○
Usability	○	○	○	○	○

Considering all factors, please select the response below that best describes your overall satisfaction level with dEv software:

- ○ Very Satisfied
- ○ Satisfied
- ○ Neutral

- Dissatisfied
- Very Dissatisfied

Appendix B.2: Test Scenarios

Test Scenario	Test Case	Steps	Expected results
Check main menu functionality	Check response of click on "+" and "- " symbols	On the main menu section 1. Click on *"+"* symbol that front of folder name 2. Click on *"-"* symbol that front of folder name	Folder will open and collapse successful
	Check response of double click on folder name	On the main menu section 1- Double click on *"Design Evaluation(dEv)"* folder	Folder content will open successfully
	Check response of click on folder name	On the main menu section 1- Double click on *"Design Evaluation(dEv)"* folder 2- Click on "focus groups" folder	Folder content will present on the centre of the screen

Test Scenario	Test Case	Steps	Expected results
Check references	Check reference open window	On the main menu section 1- Double click on *"Design Evaluation(dEv)"* folder 2- Click on "focus	Reference will open in new window

validation		groups" folder On the interface right side references will presented *1-* Click on *"Overview of qualitative methods and analytic techniques"* link.	

Appendix B.3: dEv Tool satisfaction results

1. What is your gender?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Male	9	90.0	90.0	90.0
	Female	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

2. Which age group do you belong to?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	18-24	2	20.0	20.0	20.0
	25-34	6	60.0	60.0	80.0
	35-44	2	20.0	20.0	100.0
	Total	10	100.0	100.0	

3. How many times have you used this software

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Once	9	90.0	90.0	90.0
	Twice	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

4. How do you rate your evaluation experience

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Beginner	3	30.0	30.0	30.0
	Intermediate	7	70.0	70.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Software has clear interface structure

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	6	60.0	60.0	60.0
	Agree	4	40.0	40.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Software has easy navigation

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	4	40.0	40.0	40.0
	Agree	3	30.0	30.0	70.0
	Neutral	3	30.0	30.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Menu that on the left side helps me find information easily

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	5	50.0	50.0	50.0
	Agree	5	50.0	50.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | References that are placed on the right side of every topic are helpful

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	3	30.0	30.0	30.0
	Agree	3	30.0	30.0	60.0
	Neutral	3	30.0	30.0	90.0
	Disagree	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | References encourage me to expand the topic for more information

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	3	30.0	30.0	30.0
	Agree	4	40.0	40.0	70.0
	Neutral	2	20.0	20.0	90.0
	Disagree	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Content structure helps me to clearly understand the presented topic

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	6	60.0	60.0	60.0
	Agree	4	40.0	40.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Using images is helpful to understand the topic

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	5	50.0	50.0	50.0
	Agree	5	50.0	50.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Using videos is helpful to understand the topic

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	6	60.0	60.0	60.0
	Agree	2	20.0	20.0	80.0
	Neutral	2	20.0	20.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | The videos provided reduced learning time

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	2	20.0	20.0	20.0
	Agree	4	40.0	40.0	60.0
	Neutral	3	30.0	30.0	90.0
	Disagree	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

To what extent do you agree with the following statements: | Using links through text to jump between different topics is obvious

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Agree	2	20.0	20.0	20.0
	Agree	6	60.0	60.0	80.0
	Strongly Disagree	1	10.0	10.0	90.0
	Disagree	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

please select how likely it is that you would recommend our software to a friend or colleague?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Passive: 7	2	20.0	20.0	20.0
	Passive: 8	4	40.0	40.0	60.0
	Promoter: 9	3	30.0	30.0	90.0
	Promoter: 10	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

For each of the following categories, please rate your satisfaction level with our software: | Software Interface

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Neutral	1	10.0	10.0	10.0
	Satisfied	6	60.0	60.0	70.0
	Very Satisfied	3	30.0	30.0	100.0
	Total	10	100.0	100.0	

For each of the following categories, please rate your satisfaction level with our software: | Overall Appearance

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Dissatisfied	1	10.0	10.0	10.0
	Neutral	1	10.0	10.0	20.0
	Satisfied	4	40.0	40.0	60.0
	Very Satisfied	4	40.0	40.0	100.0
	Total	10	100.0	100.0	

For each of the following categories, please rate your satisfaction level with our software: | Usability

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Dissatisfied	1	10.0	10.0	10.0
	Satisfied	3	30.0	30.0	40.0
	Very Satisfied	6	60.0	60.0	100.0
	Total	10	100.0	100.0	

Considering all factors, please select the response below that best describes your overall satisfaction level with dEv software:

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Very Satisfied	5	50.0	50.0	50.0
	Satisfied	5	50.0	50.0	100.0
	Total	10	100.0	100.0	

Appendix C.1: Clock Study Interview questions

1. Do you find the task is needed? How?

2. Which evaluation methods have you used on each stage?
    - Requirements
    - Design
    - Implement
    - Testing

3. Which of development stages do you think it's more important for developer?

4. Which of development stages do you think it's difficult to manage?

5. Does method that you used mention for any evaluation methods?

6. Does method that you used asking for specific evaluation method to use? And have you use it?

7. How many users have you involved in the solution? And which methods have you used to involve them?

8. What did you learn from this method?

9. Di you think this methods is focus on design principles, evaluation or both?

Appendix C.2: Developer Survey

Section One: Background information

1. What is your gender?

- ○ Male
- ○ Female

2. Which age group do you belong to?

- ○ 18-24

- ○ 25-34
- ○ 35-44
- ○ 45-54
- ○ 55-64
- ○ 65+

3. How much programming experience do you have?

- ○ Less than 1 year
- ○ 1 year
- ○ 2 years
- ○ 3 years
- ○ More than 3 years

4. Please select which statement best describes your occupation

- ○ I am an Undergraduate student
- ○ I am a Postgraduate student
- ○ I work full time within Education
- ○ I work full time in another area
- ○ Other, please specify... _____

5. Which method have you used for user collection requirements?

- □ Focus Groups
- □ Individual interview
- □ Developer experience
- □ Other, please specify... _____

6. On a scale of 0 – 10 with 10 being the most positive, how would you rate the software in terms of it meeting its design requirements

- ○ 10
- ○ 9
- ○ 8
- ○ 7
- ○ 6
- ○ 5
- ○ 4
- ○ 3

○ 2

○ 1

○ 0

7. How many users haveparticipated in the software requirements stage?

○ No Users

○ 1 User

○ 2 Users

○ 3 Users

○ More than 3 Users

8. How many users have participated in the software designing stage?

○ No Users

○ 1 User

○ 2 Users

○ 3 Users

○ More than 3 Users

9. How many users have participated in the software testing stage?

○ No Users

○ 1 User

○ 2 Users

○ 3 Users

○ More than 3 Users

10. What did you like most from the method?

11. What did you dislike most from the method?

10. To what extent do you agree with the following statements about the dEv Resource:

	Strongly Agree	Agree	Neutral	Strongly Disagree	Disagree
Provided useful information that is important to learn .	○	○	○	○	○
Teaching me how to do design evaluation.	○	○	○	○	○

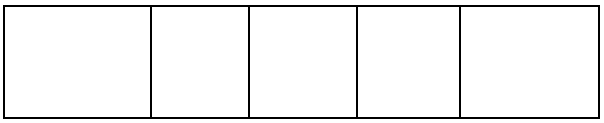
Teaching me how to involve users in software design.	○	○	○	○	○
Increases my knowledge about design evaluation.	○	○	○	○	○
Encourages me to learn more about design evaluation.	○	○	○	○	○
Recommended for Software engineers to deal with as beginning of design evaluation topic.	○	○	○	○	○

Appendix D.1: SUS Questionnaire

	Strongly Disagree				Strongly agree
1. I think that I would like to use this system frequently					
	1	2	3	4	5
2. I found the system unnecessarily complex					
	1	2	3	4	5
3. I thought the system was easy to use					
	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system					
	1	2	3	4	5
5. I found the various functions in this system were well integrated					

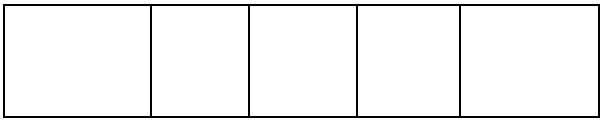
1 2 3 4 5

6. I thought there was too much inconsistency in this system



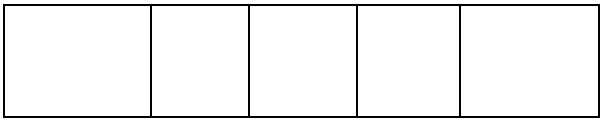
1 2 3 4 5

7. I would imagine that most people would learn to use this system very quickly



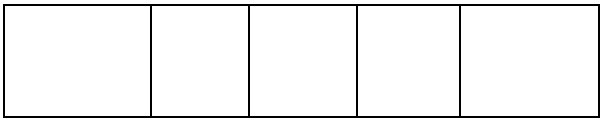
1 2 3 4 5

8. I found the system very cumbersome to use



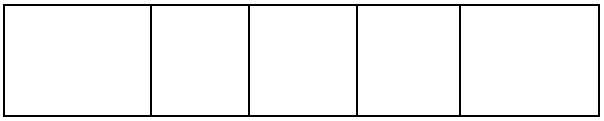
1 2 3 4 5

9. I felt very confident using the system



1 2 3 4 5

10. I needed to learn a lot of things before I could get going with this system



1 2 3 4 5

Total score =

SUS score =

Appendix D.2 SPSS Analysis and Results

Group C Web overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		90.333
Median		95.000
Mode		95.0
Minimum		77.5
Maximum		97.5
Sum		1355.0

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	2	13.3	13.3	13.3
	80.0	1	6.7	6.7	20.0
	82.5	1	6.7	6.7	26.7
	90.0	2	13.3	13.3	40.0
	92.5	1	6.7	6.7	46.7
	95.0	6	40.0	40.0	86.7
	97.5	2	13.3	13.3	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSB
V3\B_Desktop_V3.sav'.
DATASET NAME DataSet27 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

Group C Desktop overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		81.833
Median		90.000
Mode		92.5
Minimum		17.5
Maximum		100.0
Sum		1227.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	17.5	1	6.7	6.7	6.7
	57.5	1	6.7	6.7	13.3
	72.5	1	6.7	6.7	20.0
	77.5	1	6.7	6.7	26.7
	85.0	1	6.7	6.7	33.3
	87.5	2	13.3	13.3	46.7
	90.0	2	13.3	13.3	60.0
	92.5	5	33.3	33.3	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
SPLIT FILE OFF.
REGRESSION
  /DESCRIPTIVES MEAN STDDEV CORR SIG N
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT SUS_Total
  /METHOD=ENTER Gender Age Pro_Exp Occupation App_code.
```

Regression overall

Descriptive Statistics

	Mean	Std. Deviation	N
SUS_Total	86.083	15.7798	30
Gender	1.07	.254	30
Age	1.67	.661	30
Pro_Exp	2.40	.855	30
Occupation	1.70	.794	30
App_code	1.50	.509	30

Correlations

		SUS_Total	Gender	Age	Pro_Exp	Occupation	App_code
Pearson Correlation	SUS_Total	1.000	.132	.300	-.072	.206	-.274
	Gender	.132	1.000	.137	-.445	.787	.000
	Age	.300	.137	1.000	-.122	.525	-.205
	Pro_Exp	-.072	-.445	-.122	1.000	-.376	.079
	Occupation	.206	.787	.525	-.376	1.000	-.128

Sig. (1-tailed)	App_code	-.274	.000	-.205	.079	-.128	1.000
	SUS_Total	.	.243	.053	.354	.138	.071
	Gender	.243	.	.235	.007	.000	.500
	Age	.053	.235	.	.260	.001	.138
	Pro_Exp	.354	.007	.260	.	.020	.338
	Occupation	.138	.000	.001	.020	.	.250
	App_code	.071	.500	.138	.338	.250	.
N	SUS_Total	30	30	30	30	30	30
	Gender	30	30	30	30	30	30
	Age	30	30	30	30	30	30
	Pro_Exp	30	30	30	30	30	30
	Occupation	30	30	30	30	30	30
	App_code	30	30	30	30	30	30

Variables Entered/Removed[a]

Model	Variables Entered	Variables Removed	Method
1	App_code, Gender, Age, Pro_Exp, Occupation[b]	.	Enter

a. Dependent Variable: SUS_Total

b. All requested variables entered.

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.389[a]	.151	-.026	15.9810

a. Predictors: (Constant), App_code, Gender, Age, Pro_Exp, Occupation

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	1091.627	5	218.325	.855	.525[b]
	Residual	6129.415	24	255.392		
	Total	7221.042	29			

a. Dependent Variable: SUS_Total

b. Predictors: (Constant), App_code, Gender, Age, Pro_Exp, Occupation

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	74.530	26.209		2.844	.009
	Gender	13.096	23.174	.211	.565	.577
	Age	7.143	6.241	.299	1.145	.264
	Pro_Exp	.470	3.897	.025	.120	.905
	Occupation	-2.730	8.330	-.137	-.328	.746
	App_code	-7.204	6.008	-.232	-1.199	.242

a. Dependent Variable: SUS_Total

SUS web C and experience

No Experience

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		91.875
Median		95.000
Mode		95.0
Minimum		82.5
Maximum		95.0
Sum		367.5

a. Pro_Exp = No Experience

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	82.5	1	25.0	25.0	25.0
	95.0	3	75.0	75.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0

Mean	86.250
Median	86.250
Mode	77.5[b]
Minimum	77.5
Maximum	95.0
Sum	172.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	50.0	50.0	50.0
	95.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		90.556
Median		92.500
Mode		90.0[b]
Minimum		77.5
Maximum		97.5
Sum		815.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	11.1	11.1	11.1
	80.0	1	11.1	11.1	22.2
	90.0	2	22.2	22.2	44.4
	92.5	1	11.1	11.1	55.6

	95.0	2	22.2	22.2	77.8
	97.5	2	22.2	22.2	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web C and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		88.000
Median		92.500
Mode		95.0
Minimum		77.5
Maximum		95.0
Sum		440.0

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	20.0	20.0	20.0
	80.0	1	20.0	20.0	40.0
	92.5	1	20.0	20.0	60.0
	95.0	2	40.0	40.0	100.0
	Total	5	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		91.250
Median		95.000
Mode		95.0
Minimum		77.5
Maximum		97.5
Sum		730.0

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	12.5	12.5	12.5
	82.5	1	12.5	12.5	25.0
	90.0	1	12.5	12.5	37.5
	95.0	3	37.5	37.5	75.0
	97.5	2	25.0	25.0	100.0
	Total	8	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		92.500
Median		92.500
Mode		90.0[b]
Minimum		90.0
Maximum		95.0
Sum		185.0

a. Age = 35-44

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	90.0	1	50.0	50.0	50.0
	95.0	1	50.0	50.0	100.0

	Total	2	100.0	100.0	

a. Age = 35-44

SUS web C and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	14
	Missing	0
Mean		90.000
Median		93.750
Mode		95.0
Std. Deviation		7.4032
Minimum		77.5
Maximum		97.5
Sum		1260.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	2	14.3	14.3	14.3
	80.0	1	7.1	7.1	21.4
	82.5	1	7.1	7.1	28.6
	90.0	2	14.3	14.3	42.9
	92.5	1	7.1	7.1	50.0
	95.0	5	35.7	35.7	85.7
	97.5	2	14.3	14.3	100.0
	Total	14	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		95.000
Median		95.000
Mode		95.0

Minimum	95.0
Maximum	95.0
Sum	95.0

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	95.0	1	100.0	100.0	100.0

a. Gender = Female

SUS web C and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		88.000
Median		92.500
Mode		95.0
Std. Deviation		8.5513
Minimum		77.5
Maximum		95.0
Sum		440.0

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	20.0	20.0	20.0
	80.0	1	20.0	20.0	40.0
	92.5	1	20.0	20.0	60.0
	95.0	2	40.0	40.0	100.0
	Total	5	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		91.111
Median		95.000
Mode		95.0
Std. Deviation		6.9722
Minimum		77.5
Maximum		97.5
Sum		820.0

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	11.1	11.1	11.1
	82.5	1	11.1	11.1	22.2
	90.0	2	22.2	22.2	44.4
	95.0	3	33.3	33.3	77.8
	97.5	2	22.2	22.2	100.0
	Total	9	100.0	100.0	

a. Occupation = 2

Occupation = Full-time employee in other area

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		95.000
Median		95.000
Mode		95.0
Minimum		95.0
Maximum		95.0
Sum		95.0

a. Occupation = 4

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

Valid	95.0	1	100.0	100.0	100.0

a. Occupation = 4

```
DATASET ACTIVATE DataSet31.
DATASET CLOSE DataSet38.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSB
V3\B_Desktop_V3.sav'.
DATASET NAME DataSet39 WINDOW=FRONT.
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

Group C Desktop and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		86.667
Median		90.000
Mode		77.5[b]
Minimum		77.5
Maximum		92.5
Sum		260.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	33.3	33.3	33.3
	90.0	1	33.3	33.3	66.7
	92.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		75.000
Median		75.000
Mode		57.5[b]
Minimum		57.5
Maximum		92.5
Sum		150.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	50.0	50.0	50.0
	92.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		81.750
Median		88.750
Mode		92.5
Minimum		17.5
Maximum		100.0
Sum		817.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	17.5	1	10.0	10.0	10.0
	72.5	1	10.0	10.0	20.0
	85.0	1	10.0	10.0	30.0
	87.5	2	20.0	20.0	50.0
	90.0	1	10.0	10.0	60.0
	92.5	3	30.0	30.0	90.0
	100.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

Group C Desktop and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		75.313
Median		88.750
Mode		92.5
Minimum		17.5
Maximum		92.5
Sum		602.5

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	17.5	1	12.5	12.5	12.5
	57.5	1	12.5	12.5	25.0
	72.5	1	12.5	12.5	37.5
	87.5	1	12.5	12.5	50.0
	90.0	1	12.5	12.5	62.5

	92.5	3	37.5	37.5	100.0
	Total	8	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		89.583
Median		91.250
Mode		92.5
Minimum		77.5
Maximum		100.0
Sum		537.5

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	16.7	16.7	16.7
	85.0	1	16.7	16.7	33.3
	90.0	1	16.7	16.7	50.0
	92.5	2	33.3	33.3	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		87.500
Median		87.500
Mode		87.5
Minimum		87.5
Maximum		87.5
Sum		87.5

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	87.5	1	100.0	100.0	100.0

a. Age = 35-44

Group C Desktop and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	14
	Missing	0
Mean		81.071
Median		88.750
Mode		92.5
Std. Deviation		21.1613
Minimum		17.5
Maximum		100.0
Sum		1135.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	17.5	1	7.1	7.1	7.1
	57.5	1	7.1	7.1	14.3
	72.5	1	7.1	7.1	21.4
	77.5	1	7.1	7.1	28.6
	85.0	1	7.1	7.1	35.7
	87.5	2	14.3	14.3	50.0
	90.0	2	14.3	14.3	64.3
	92.5	4	28.6	28.6	92.9
	100.0	1	7.1	7.1	100.0
	Total	14	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	1

	Missing	0
Mean		92.500
Median		92.500
Mode		92.5
Minimum		92.5
Maximum		92.5
Sum		92.5

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	92.5	1	100.0	100.0	100.0

a. Gender = Female

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

Group C Desktop and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		79.375
Median		90.000
Mode		92.5
Std. Deviation		25.8688
Minimum		17.5
Maximum		92.5
Sum		635.0

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	17.5	1	12.5	12.5	12.5
	72.5	1	12.5	12.5	25.0
	87.5	1	12.5	12.5	37.5
	90.0	2	25.0	25.0	62.5
	92.5	3	37.5	37.5	100.0
	Total	8	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		83.333
Median		86.250
Mode		57.5[b]
Std. Deviation		14.7196
Minimum		57.5
Maximum		100.0
Sum		500.0

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	16.7	16.7	16.7
	77.5	1	16.7	16.7	33.3
	85.0	1	16.7	16.7	50.0
	87.5	1	16.7	16.7	66.7
	92.5	1	16.7	16.7	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full-time employee in other area

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		92.500
Median		92.500
Mode		92.5
Minimum		92.5
Maximum		92.5
Sum		92.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	92.5	1	100.0	100.0	100.0

a. Occupation = 4

```
DATASET ACTIVATE DataSet12.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for C Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	23.333	1	23.333	.426	.525
Within Groups	712.500	13	54.808		
Total	735.833	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	43.333	2	21.667	.375	.695
Within Groups	692.500	12	57.708		
Total	735.833	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	43.299	2	21.649	.375	.695
Within Groups	692.535	12	57.711		
Total	735.833	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	54.444	2	27.222	.479	.631
Within Groups	681.389	12	56.782		
Total	735.833	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet12.
DATASET ACTIVATE DataSet13.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop application
Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	121.905	1	121.905	.272	.611
Within Groups	5821.429	13	447.802		
Total	5943.333	14			

```
ONEWAY SUS_Total BY Age
```

```
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	732.656	2	366.328	.844	.454
Within Groups	5210.677	12	434.223		
Total	5943.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	163.542	2	81.771	.170	.846
Within Groups	5779.792	12	481.649		
Total	5943.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	175.625	2	87.813	.183	.835
Within Groups	5767.708	12	480.642		
Total	5943.333	14			

Group D Web overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		67.167

Median	70.000
Mode	72.5
Minimum	35.0
Maximum	87.5
Sum	1007.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	6.7	6.7	6.7
	45.0	1	6.7	6.7	13.3
	60.0	1	6.7	6.7	20.0
	62.5	1	6.7	6.7	26.7
	65.0	1	6.7	6.7	33.3
	67.5	2	13.3	13.3	46.7
	70.0	1	6.7	6.7	53.3
	72.5	3	20.0	20.0	73.3
	75.0	1	6.7	6.7	80.0
	77.5	2	13.3	13.3	93.3
	87.5	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSD
V3\D_Desktop_V3.sav'.
DATASET NAME DataSet30 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

Group D Desktop Overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		82.833
Median		85.000
Mode		75.0[a]

Minimum	57.5
Maximum	100.0
Sum	1242.5

a. Multiple modes exist. The smallest value is shown

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	6.7	6.7	6.7
	70.0	1	6.7	6.7	13.3
	75.0	2	13.3	13.3	26.7
	77.5	1	6.7	6.7	33.3
	82.5	2	13.3	13.3	46.7
	85.0	2	13.3	13.3	60.0
	87.5	2	13.3	13.3	73.3
	90.0	1	6.7	6.7	80.0
	92.5	1	6.7	6.7	86.7
	95.0	1	6.7	6.7	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

SUS web D and experience

No Experience

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		60.000
Median		58.750
Mode		35.0[b]
Minimum		35.0
Maximum		87.5
Sum		240.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	25.0	25.0	25.0
	45.0	1	25.0	25.0	50.0
	72.5	1	25.0	25.0	75.0
	87.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		72.000
Median		72.500
Mode		67.5
Minimum		67.5
Maximum		77.5
Sum		360.0

a. Pro_Exp = Year or less than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	67.5	2	40.0	40.0	40.0
	72.5	1	20.0	20.0	60.0
	75.0	1	20.0	20.0	80.0
	77.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	6

	Missing	0
Mean		67.917
Median		67.500
Mode		60.0[b]
Minimum		60.0
Maximum		77.5
Sum		407.5

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	60.0	1	16.7	16.7	16.7
	62.5	1	16.7	16.7	33.3
	65.0	1	16.7	16.7	50.0
	70.0	1	16.7	16.7	66.7
	72.5	1	16.7	16.7	83.3
	77.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web D and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		68.333
Median		72.500
Mode		72.5
Std. Deviation		14.5237
Minimum		35.0
Maximum		87.5

Sum	615.0

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	11.1	11.1	11.1
	62.5	1	11.1	11.1	22.2
	65.0	1	11.1	11.1	33.3
	67.5	1	11.1	11.1	44.4
	72.5	2	22.2	22.2	66.7
	75.0	1	11.1	11.1	77.8
	77.5	1	11.1	11.1	88.9
	87.5	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		65.417
Median		68.750
Mode		45.0[b]
Std. Deviation		11.5560
Minimum		45.0
Maximum		77.5
Sum		392.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	16.7	16.7	16.7
	60.0	1	16.7	16.7	33.3
	67.5	1	16.7	16.7	50.0

	70.0	1	16.7	16.7	66.7
	72.5	1	16.7	16.7	83.3
	77.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

SUS web D and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		66.731
Median		70.000
Mode		72.5[b]
Std. Deviation		14.0084
Minimum		35.0
Maximum		87.5
Sum		867.5

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	7.7	7.7	7.7
	45.0	1	7.7	7.7	15.4
	60.0	1	7.7	7.7	23.1
	62.5	1	7.7	7.7	30.8
	65.0	1	7.7	7.7	38.5
	67.5	1	7.7	7.7	46.2
	70.0	1	7.7	7.7	53.8
	72.5	2	15.4	15.4	69.2
	75.0	1	7.7	7.7	76.9

	77.5	2	15.4	15.4	92.3
	87.5	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		70.000
Median		70.000
Mode		67.5[b]
Std. Deviation		3.5355
Minimum		67.5
Maximum		72.5
Sum		140.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	67.5	1	50.0	50.0	50.0
	72.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

SUS web D and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		66.111
Median		67.500
Mode		67.5[b]
Minimum		35.0
Maximum		77.5
Sum		595.0

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	11.1	11.1	11.1
	62.5	1	11.1	11.1	22.2
	65.0	1	11.1	11.1	33.3
	67.5	2	22.2	22.2	55.6
	72.5	2	22.2	22.2	77.8
	75.0	1	11.1	11.1	88.9
	77.5	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		68.750
Median		71.250
Mode		45.0[b]
Minimum		45.0
Maximum		87.5
Sum		275.0

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	25.0	25.0	25.0
	70.0	1	25.0	25.0	50.0
	72.5	1	25.0	25.0	75.0
	87.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		68.750
Median		68.750
Mode		60.0[b]
Minimum		60.0
Maximum		77.5
Sum		137.5

a. Age = 35-44

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	60.0	1	50.0	50.0	50.0
	77.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Age = 35-44

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSD
V3\D_Desktop_V3.sav'.
DATASET NAME DataSet37 WINDOW=FRONT.
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop D and experience

No Experience

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		78.750
Median		78.750
Mode		57.5[b]
Minimum		57.5
Maximum		100.0
Sum		472.5

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	16.7	16.7	16.7
	70.0	1	16.7	16.7	33.3
	75.0	1	16.7	16.7	50.0
	82.5	1	16.7	16.7	66.7
	87.5	1	16.7	16.7	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		83.333
Median		85.000

Mode	75.0[b]
Minimum	75.0
Maximum	90.0
Sum	250.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	1	33.3	33.3	33.3
	85.0	1	33.3	33.3	66.7
	90.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		86.667
Median		86.250
Mode		77.5[b]
Minimum		77.5
Maximum		95.0
Sum		520.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	16.7	16.7	16.7
	82.5	1	16.7	16.7	33.3

	85.0	1	16.7	16.7	50.0
	87.5	1	16.7	16.7	66.7
	92.5	1	16.7	16.7	83.3
	95.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = More than 1 year

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSD
V3\D_Desktop_V3.sav'.
DATASET NAME DataSet15 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop D Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		83.077
Median		85.000
Mode		82.5[b]
Std. Deviation		11.2340
Minimum		57.5
Maximum		100.0
Sum		1080.0

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	7.7	7.7	7.7
	70.0	1	7.7	7.7	15.4
	75.0	1	7.7	7.7	23.1

	77.5	1	7.7	7.7	30.8
	82.5	2	15.4	15.4	46.2
	85.0	2	15.4	15.4	61.5
	87.5	1	7.7	7.7	69.2
	90.0	1	7.7	7.7	76.9
	92.5	1	7.7	7.7	84.6
	95.0	1	7.7	7.7	92.3
	100.0	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		81.250
Median		81.250
Mode		75.0[b]
Std. Deviation		8.8388
Minimum		75.0
Maximum		87.5
Sum		162.5

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	1	50.0	50.0	50.0
	87.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop D Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		82.813
Median		85.000
Mode		85.0
Std. Deviation		13.1229
Minimum		57.5
Maximum		100.0
Sum		662.5

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	12.5	12.5	12.5
	75.0	1	12.5	12.5	25.0
	77.5	1	12.5	12.5	37.5
	85.0	2	25.0	25.0	62.5
	87.5	1	12.5	12.5	75.0
	95.0	1	12.5	12.5	87.5
	100.0	1	12.5	12.5	100.0
	Total	8	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		82.083
Median		82.500
Mode		82.5
Std. Deviation		8.5756
Minimum		70.0

Maximum	92.5
Sum	492.5

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	16.7	16.7	16.7
	75.0	1	16.7	16.7	33.3
	82.5	2	33.3	33.3	66.7
	90.0	1	16.7	16.7	83.3
	92.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		87.500
Median		87.500
Mode		87.5
Minimum		87.5
Maximum		87.5
Sum		87.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	87.5	1	100.0	100.0	100.0

a. Occupation = 4

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop D and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		80.357
Median		85.000
Mode		85.0
Std. Deviation		12.0268
Minimum		57.5
Maximum		95.0
Sum		562.5

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	14.3	14.3	14.3
	75.0	1	14.3	14.3	28.6
	77.5	1	14.3	14.3	42.9
	85.0	2	28.6	28.6	71.4
	87.5	1	14.3	14.3	85.7
	95.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		82.917
Median		82.500
Mode		82.5
Std. Deviation		10.4183
Minimum		70.0
Maximum		100.0
Sum		497.5

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	16.7	16.7	16.7
	75.0	1	16.7	16.7	33.3
	82.5	2	33.3	33.3	66.7
	87.5	1	16.7	16.7	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		91.250
Median		91.250
Mode		90.0[b]
Std. Deviation		1.7678
Minimum		90.0
Maximum		92.5
Sum		182.5

a. Age = 35-44

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	90.0	1	50.0	50.0	50.0
	92.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Age = 35-44

```
DATASET ACTIVATE DataSet14.
SPLIT FILE OFF.
```

```
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for D Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	18.526	1	18.526	.102	.755
Within Groups	2367.308	13	182.101		
Total	2385.833	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	25.069	2	12.535	.064	.939
Within Groups	2360.764	12	196.730		
Total	2385.833	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Programing Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	325.625	2	162.813	.948	.415
Within Groups	2060.208	12	171.684		
Total	2385.833	14			

```
ONEWAY SUS_Total BY Occupation
```

```
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	30.625	1	30.625	.169	.688
Within Groups	2355.208	13	181.170		
Total	2385.833	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet14.
DATASET ACTIVATE DataSet15.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	5.785	1	5.785	.047	.831
Within Groups	1592.548	13	122.504		
Total	1598.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	184.643	2	92.321	.784	.479
Within Groups	1413.690	12	117.808		
Total	1598.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Programing Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	188.958	2	94.479	.804	.470
Within Groups	1409.375	12	117.448		
Total	1598.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	25.156	2	12.578	.096	.909
Within Groups	1573.177	12	131.098		
Total	1598.333	14			

SUS Web F overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		46.500
Median		42.500
Mode		30.0[a]
Minimum		15.0

Maximum	92.5
Sum	697.5

a. Multiple modes exist. The smallest value is shown

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	6.7	6.7	6.7
	22.5	1	6.7	6.7	13.3
	30.0	2	13.3	13.3	26.7
	32.5	1	6.7	6.7	33.3
	35.0	1	6.7	6.7	40.0
	40.0	1	6.7	6.7	46.7
	42.5	2	13.3	13.3	60.0
	52.5	1	6.7	6.7	66.7
	60.0	2	13.3	13.3	80.0
	62.5	1	6.7	6.7	86.7
	80.0	1	6.7	6.7	93.3
	92.5	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Desktop_V3.sav'.
DATASET NAME DataSet33 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop F overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		68.333
Median		67.500
Mode		30.0[a]
Minimum		30.0

Maximum	100.0
Sum	1025.0

a. Multiple modes exist. The smallest value is shown

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	2	13.3	13.3	13.3
	50.0	1	6.7	6.7	20.0
	52.5	1	6.7	6.7	26.7
	57.5	2	13.3	13.3	40.0
	62.5	1	6.7	6.7	46.7
	67.5	1	6.7	6.7	53.3
	75.0	1	6.7	6.7	60.0
	77.5	2	13.3	13.3	73.3
	95.0	2	13.3	13.3	86.7
	97.5	1	6.7	6.7	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
SPLIT FILE OFF.
REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT SUS_Total
  /METHOD=ENTER Gender Age Pro_Exp Occupation Framework App_code.
```

SUS Desktop F and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		37.000
Median		35.000

Mode	15.0[b]
Minimum	15.0
Maximum	62.5
Sum	185.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		45.500
Median		42.500
Mode		32.5[b]
Minimum		32.5
Maximum		60.0
Sum		227.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	32.5	1	20.0	20.0	20.0
	40.0	1	20.0	20.0	40.0
	42.5	1	20.0	20.0	60.0
	52.5	1	20.0	20.0	80.0
	60.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		57.000
Median		60.000
Mode		22.5[b]
Minimum		22.5
Maximum		92.5
Sum		285.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	20.0	20.0	20.0
	30.0	1	20.0	20.0	40.0
	60.0	1	20.0	20.0	60.0
	80.0	1	20.0	20.0	80.0
	92.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	20.0	20.0	20.0
	30.0	1	20.0	20.0	40.0
	35.0	1	20.0	20.0	60.0
	42.5	1	20.0	20.0	80.0
	62.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = No Experience

```
DATASET ACTIVATE DataSet31.
DATASET CLOSE DataSet34.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Desktop_V3.sav'.
DATASET NAME DataSet35 WINDOW=FRONT.
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop F and experience

No Experience

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		66.000
Median		62.500
Mode		50.0[b]
Minimum		50.0
Maximum		97.5
Sum		330.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	50.0	1	20.0	20.0	20.0
	52.5	1	20.0	20.0	40.0
	62.5	1	20.0	20.0	60.0
	67.5	1	20.0	20.0	80.0
	97.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		58.000
Median		57.500
Mode		30.0
Minimum		30.0
Maximum		95.0
Sum		290.0

a. Pro_Exp = Year or less than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	2	40.0	40.0	40.0
	57.5	1	20.0	20.0	60.0
	77.5	1	20.0	20.0	80.0
	95.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		81.000
Median		77.500
Mode		57.5[b]
Minimum		57.5
Maximum		100.0

Sum	405.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	57.5	1	20.0	20.0	20.0
	75.0	1	20.0	20.0	40.0
	77.5	1	20.0	20.0	60.0
	95.0	1	20.0	20.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = More than 1 year

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Web_V3.sav'.
DATASET NAME DataSet38 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web F and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		46.111
Median		40.000
Mode		22.5[b]
Std. Deviation		18.7546

Minimum	22.5
Maximum	80.0
Sum	415.0

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	11.1	11.1	11.1
	30.0	1	11.1	11.1	22.2
	32.5	1	11.1	11.1	33.3
	35.0	1	11.1	11.1	44.4
	40.0	1	11.1	11.1	55.6
	52.5	1	11.1	11.1	66.7
	60.0	1	11.1	11.1	77.8
	62.5	1	11.1	11.1	88.9
	80.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		38.000
Median		42.500
Mode		42.5
Std. Deviation		16.7145
Minimum		15.0
Maximum		60.0
Sum		190.0

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	20.0	20.0	20.0
	30.0	1	20.0	20.0	40.0
	42.5	2	40.0	40.0	80.0
	60.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		92.500
Median		92.500
Mode		92.5
Minimum		92.5
Maximum		92.5
Sum		92.5

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	92.5	1	100.0	100.0	100.0

a. Age = 35-44

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Desktop_V3.sav'.
DATASET NAME DataSet39 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop F and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		64.167
Median		57.500
Mode		57.5[b]
Std. Deviation		20.4634
Minimum		30.0
Maximum		100.0
Sum		577.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	11.1	11.1	11.1
	50.0	1	11.1	11.1	22.2
	52.5	1	11.1	11.1	33.3
	57.5	2	22.2	22.2	55.6
	75.0	1	11.1	11.1	66.7
	77.5	2	22.2	22.2	88.9
	100.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		70.500
Median		67.500

Mode	30.0[b]
Std. Deviation	27.5794
Minimum	30.0
Maximum	97.5
Sum	352.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	20.0	20.0	20.0
	62.5	1	20.0	20.0	40.0
	67.5	1	20.0	20.0	60.0
	95.0	1	20.0	20.0	80.0
	97.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		95.000
Median		95.000
Mode		95.0
Minimum		95.0
Maximum		95.0
Sum		95.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	95.0	1	100.0	100.0	100.0

a. Age = 35-44

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Web_V3.sav'.
DATASET NAME DataSet18 WINDOW=FRONT.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSE
V3\E_Desktop_V3.sav'.
DATASET NAME DataSet19 WINDOW=FRONT.
DATASET ACTIVATE DataSet18.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web F and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0
Mean		44.375
Median		37.500
Mode		30.0
Std. Deviation		23.1626
Minimum		15.0
Maximum		92.5
Sum		532.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	8.3	8.3	8.3
	22.5	1	8.3	8.3	16.7
	30.0	2	16.7	16.7	33.3

	32.5	1	8.3	8.3	41.7
	35.0	1	8.3	8.3	50.0
	40.0	1	8.3	8.3	58.3
	42.5	1	8.3	8.3	66.7
	52.5	1	8.3	8.3	75.0
	60.0	1	8.3	8.3	83.3
	80.0	1	8.3	8.3	91.7
	92.5	1	8.3	8.3	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		55.000
Median		60.000
Mode		42.5[b]
Std. Deviation		10.8972
Minimum		42.5
Maximum		62.5
Sum		165.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	33.3	33.3	33.3
	60.0	1	33.3	33.3	66.7
	62.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
DATASET ACTIVATE DataSet19.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop F and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0
Mean		73.542
Median		76.250
Mode		57.5[b]
Std. Deviation		21.5707
Minimum		30.0
Maximum		100.0
Sum		882.5

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	8.3	8.3	8.3
	52.5	1	8.3	8.3	16.7
	57.5	2	16.7	16.7	33.3
	67.5	1	8.3	8.3	41.7
	75.0	1	8.3	8.3	50.0
	77.5	2	16.7	16.7	66.7
	95.0	2	16.7	16.7	83.3
	97.5	1	8.3	8.3	91.7
	100.0	1	8.3	8.3	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		47.500
Median		50.000
Mode		30.0[b]
Std. Deviation		16.3936
Minimum		30.0
Maximum		62.5
Sum		142.5

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	33.3	33.3	33.3
	50.0	1	33.3	33.3	66.7
	62.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
DATASET ACTIVATE DataSet18.
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web F and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		47.500
Median		46.250
Mode		60.0

Std. Deviation	18.2193
Minimum	22.5
Maximum	80.0
Sum	475.0

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	10.0	10.0	10.0
	30.0	1	10.0	10.0	20.0
	32.5	1	10.0	10.0	30.0
	35.0	1	10.0	10.0	40.0
	40.0	1	10.0	10.0	50.0
	52.5	1	10.0	10.0	60.0
	60.0	2	20.0	20.0	80.0
	62.5	1	10.0	10.0	90.0
	80.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		45.000
Median		36.250
Mode		15.0[b]
Std. Deviation		33.6031
Minimum		15.0
Maximum		92.5
Sum		180.0

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	25.0	25.0	25.0
	30.0	1	25.0	25.0	50.0
	42.5	1	25.0	25.0	75.0
	92.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		42.500
Median		42.500
Mode		42.5
Minimum		42.5
Maximum		42.5
Sum		42.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	100.0	100.0	100.0

a. Occupation = 4

```
DATASET ACTIVATE DataSet19.
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop F and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		60.750
Median		57.500
Mode		30.0[b]
Std. Deviation		22.1124
Minimum		30.0
Maximum		100.0
Sum		607.5

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	2	20.0	20.0	20.0
	50.0	1	10.0	10.0	30.0
	52.5	1	10.0	10.0	40.0
	57.5	2	20.0	20.0	60.0
	75.0	1	10.0	10.0	70.0
	77.5	2	20.0	20.0	90.0
	100.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		88.750
Median		95.000
Mode		95.0
Std. Deviation		14.2156
Minimum		67.5
Maximum		97.5

Sum	355.0

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	67.5	1	25.0	25.0	25.0
	95.0	2	50.0	50.0	75.0
	97.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		62.500
Median		62.500
Mode		62.5
Minimum		62.5
Maximum		62.5
Sum		62.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	62.5	1	100.0	100.0	100.0

a. Occupation = 4

```
DATASET ACTIVATE DataSet18.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for F Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	270.938	1	270.938	.574	.462
Within Groups	6139.063	13	472.236		
Total	6410.000	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2478.611	2	1239.306	3.783	.053
Within Groups	3931.389	12	327.616		
Total	6410.000	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1007.500	2	503.750	1.119	.358
Within Groups	5402.500	12	450.208		
Total	6410.000	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.

Between Groups	35.000	2	17.500	.033	.968
Within Groups	6375.000	12	531.250		
Total	6410.000	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet18.
DATASET ACTIVATE DataSet19.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1627.604	1	1627.604	3.741	.075
Within Groups	5655.729	13	435.056		
Total	7283.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	890.833	2	445.417	.836	.457
Within Groups	6392.500	12	532.708		
Total	7283.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1363.333	2	681.667	1.382	.288
Within Groups	5920.000	12	493.333		
Total	7283.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2276.458	2	1138.229	2.728	.106
Within Groups	5006.875	12	417.240		
Total	7283.333	14			

SUS Web G overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		67.833
Median		75.000
Mode		70.0[b]
Minimum		25.0
Maximum		90.0
Sum		1017.5

a. App_code = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	6.7	6.7	6.7
	35.0	1	6.7	6.7	13.3
	45.0	1	6.7	6.7	20.0
	55.0	1	6.7	6.7	26.7
	57.5	1	6.7	6.7	33.3
	70.0	2	13.3	13.3	46.7
	75.0	1	6.7	6.7	53.3
	77.5	2	13.3	13.3	66.7
	80.0	1	6.7	6.7	73.3
	82.5	1	6.7	6.7	80.0
	87.5	1	6.7	6.7	86.7
	90.0	2	13.3	13.3	100.0
	Total	15	100.0	100.0	

a. App_code = 1

SUS Desktop G overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		73.667
Median		82.500
Mode		92.5
Minimum		15.0
Maximum		97.5
Sum		1105.0

a. App_code = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	6.7	6.7	6.7
	47.5	1	6.7	6.7	13.3
	50.0	1	6.7	6.7	20.0
	52.5	1	6.7	6.7	26.7
	62.5	1	6.7	6.7	33.3
	72.5	1	6.7	6.7	40.0
	75.0	1	6.7	6.7	46.7
	82.5	1	6.7	6.7	53.3
	87.5	1	6.7	6.7	60.0
	90.0	1	6.7	6.7	66.7
	92.5	3	20.0	20.0	86.7
	95.0	1	6.7	6.7	93.3
	97.5	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

a. App_code = 2

```
SPLIT FILE OFF.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSF
V3\F_Web_V3.sav'.
DATASET NAME DataSet4 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.


GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSF
V3\F_Web_V3.sav'.
DATASET NAME DataSet20 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web G and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		68.654
Median		75.000
Mode		70.0[b]
Std. Deviation		18.5016
Minimum		25.0
Maximum		90.0
Sum		892.5

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	7.7	7.7	7.7
	45.0	1	7.7	7.7	15.4
	55.0	1	7.7	7.7	23.1
	57.5	1	7.7	7.7	30.8
	70.0	2	15.4	15.4	46.2
	75.0	1	7.7	7.7	53.8
	77.5	2	15.4	15.4	69.2
	80.0	1	7.7	7.7	76.9
	82.5	1	7.7	7.7	84.6
	87.5	1	7.7	7.7	92.3
	90.0	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0

Mean	62.500
Median	62.500
Mode	35.0[b]
Std. Deviation	38.8909
Minimum	35.0
Maximum	90.0
Sum	125.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	50.0	50.0	50.0
	90.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web G and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		78.889
Median		80.000
Mode		77.5[b]
Std. Deviation		11.1181
Minimum		55.0
Maximum		90.0

Sum	710.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	55.0	1	11.1	11.1	11.1
	70.0	1	11.1	11.1	22.2
	77.5	2	22.2	22.2	44.4
	80.0	1	11.1	11.1	55.6
	82.5	1	11.1	11.1	66.7
	87.5	1	11.1	11.1	77.8
	90.0	2	22.2	22.2	100.0
	Total	9	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		51.250
Median		51.250
Mode		25.0[b]
Std. Deviation		19.7326
Minimum		25.0
Maximum		75.0
Sum		307.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	16.7	16.7	16.7
	35.0	1	16.7	16.7	33.3
	45.0	1	16.7	16.7	50.0
	57.5	1	16.7	16.7	66.7
	70.0	1	16.7	16.7	83.3
	75.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

SUS web G and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		70.625
Median		77.500
Mode		77.5
Minimum		35.0
Maximum		87.5
Sum		565.0

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	12.5	12.5	12.5
	55.0	1	12.5	12.5	25.0
	70.0	1	12.5	12.5	37.5
	77.5	2	25.0	25.0	62.5
	80.0	1	12.5	12.5	75.0
	82.5	1	12.5	12.5	87.5
	87.5	1	12.5	12.5	100.0
	Total	8	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		60.417
Median		63.750
Mode		25.0[b]
Minimum		25.0
Maximum		90.0
Sum		362.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	16.7	16.7	16.7
	45.0	1	16.7	16.7	33.3
	57.5	1	16.7	16.7	50.0
	70.0	1	16.7	16.7	66.7
	75.0	1	16.7	16.7	83.3
	90.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 25-34

Age = 55-64

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		90.000

Median	90.000
Mode	90.0
Minimum	90.0
Maximum	90.0
Sum	90.0

a. Age = 55-64

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	90.0	1	100.0	100.0	100.0

a. Age = 55-64

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web G and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		35.000
Median		35.000
Mode		25.0[b]
Minimum		25.0
Maximum		45.0
Sum		70.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	50.0	50.0	50.0

	45.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		84.000
Median		82.500
Mode		90.0
Minimum		77.5
Maximum		90.0
Sum		420.0

a. Pro_Exp = Year or less than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	20.0	20.0	20.0
	80.0	1	20.0	20.0	40.0
	82.5	1	20.0	20.0	60.0
	90.0	2	40.0	40.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		65.938

Median	70.000
Mode	70.0
Minimum	35.0
Maximum	87.5
Sum	527.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	12.5	12.5	12.5
	55.0	1	12.5	12.5	25.0
	57.5	1	12.5	12.5	37.5
	70.0	2	25.0	25.0	62.5
	75.0	1	12.5	12.5	75.0
	77.5	1	12.5	12.5	87.5
	87.5	1	12.5	12.5	100.0
	Total	8	100.0	100.0	

a. Pro_Exp = More than 1 year

```
DATASET ACTIVATE DataSet3.
DATASET CLOSE DataSet4.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSF
V3\F_Desktop_V3.sav'.
DATASET NAME DataSet5 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop G and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		80.357
Median		87.500

Mode	52.5[b]
Minimum	52.5
Maximum	95.0
Sum	562.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	14.3	14.3	14.3
	62.5	1	14.3	14.3	28.6
	82.5	1	14.3	14.3	42.9
	87.5	1	14.3	14.3	57.1
	90.0	1	14.3	14.3	71.4
	92.5	1	14.3	14.3	85.7
	95.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		72.500
Median		73.750
Mode		47.5[b]
Minimum		47.5
Maximum		97.5
Sum		435.0

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	47.5	1	16.7	16.7	16.7
	50.0	1	16.7	16.7	33.3
	72.5	1	16.7	16.7	50.0
	75.0	1	16.7	16.7	66.7
	92.5	1	16.7	16.7	83.3
	97.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		92.500
Median		92.500
Mode		92.5
Minimum		92.5
Maximum		92.5
Sum		92.5

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	92.5	1	100.0	100.0	100.0

a. Age = 35-44

Age = 55-64

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		15.000

Median	15.000
Mode	15.0
Minimum	15.0
Maximum	15.0
Sum	15.0

a. Age = 55-64

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	100.0	100.0	100.0

a. Age = 55-64

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop G and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		73.750
Median		73.750
Mode		50.0[b]
Minimum		50.0
Maximum		97.5
Sum		147.5

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

	50.0	1	50.0	50.0	50.0
Valid	97.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		70.417
Median		80.000
Mode		92.5
Minimum		15.0
Maximum		92.5
Sum		422.5

a. Pro_Exp = Year or less than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	16.7	16.7	16.7
	62.5	1	16.7	16.7	33.3
	72.5	1	16.7	16.7	50.0
	87.5	1	16.7	16.7	66.7
	92.5	2	33.3	33.3	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		76.429
Median		82.500
Mode		47.5[b]

Minimum	47.5
Maximum	95.0
Sum	535.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	47.5	1	14.3	14.3	14.3
	52.5	1	14.3	14.3	28.6
	75.0	1	14.3	14.3	42.9
	82.5	1	14.3	14.3	57.1
	90.0	1	14.3	14.3	71.4
	92.5	1	14.3	14.3	85.7
	95.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop G and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	14
	Missing	0
Mean		73.750
Median		85.000
Mode		92.5
Minimum		15.0
Maximum		97.5
Sum		1032.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	7.1	7.1	7.1
	47.5	1	7.1	7.1	14.3
	50.0	1	7.1	7.1	21.4
	52.5	1	7.1	7.1	28.6
	62.5	1	7.1	7.1	35.7
	75.0	1	7.1	7.1	42.9
	82.5	1	7.1	7.1	50.0
	87.5	1	7.1	7.1	57.1
	90.0	1	7.1	7.1	64.3
	92.5	3	21.4	21.4	85.7
	95.0	1	7.1	7.1	92.9
	97.5	1	7.1	7.1	100.0
	Total	14	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		72.500
Median		72.500
Mode		72.5
Minimum		72.5
Maximum		72.5
Sum		72.5

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	1	100.0	100.0	100.0

a. Gender = Female

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSF
V3\F_Desktop_V3.sav'.
DATASET NAME DataSet21 WINDOW=FRONT.
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop G Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		72.222
Median		82.500
Mode		15.0[b]
Std. Deviation		25.8434
Minimum		15.0
Maximum		95.0
Sum		650.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	15.0	1	11.1	11.1	11.1
	52.5	1	11.1	11.1	22.2
	62.5	1	11.1	11.1	33.3
	72.5	1	11.1	11.1	44.4
	82.5	1	11.1	11.1	55.6
	87.5	1	11.1	11.1	66.7
	90.0	1	11.1	11.1	77.8
	92.5	1	11.1	11.1	88.9

	95.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		75.833
Median		83.750
Mode		92.5
Std. Deviation		22.3420
Minimum		47.5
Maximum		97.5
Sum		455.0

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	47.5	1	16.7	16.7	16.7
	50.0	1	16.7	16.7	33.3
	75.0	1	16.7	16.7	50.0
	92.5	2	33.3	33.3	83.3
	97.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

```
DATASET ACTIVATE DataSet20.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for G Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	65.641	1	65.641	.152	.703
Within Groups	5620.192	13	432.322		
Total	5685.833	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

_V3.sav

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	883.750	2	441.875	1.104	.363
Within Groups	4802.083	12	400.174		
Total	5685.833	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	3491.615	2	1745.807	9.548	.003
Within Groups	2194.219	12	182.852		
Total	5685.833	14			

Descriptives

SUS_Total

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
No Experience	2	35.000	14.1421	10.0000	-92.062	162.062	25.0	45.0

Year or less than 1 year	5	84.000	5.7554	2.5739	76.854	91.146	77.5	90.0
More than 1 year	8	65.938	16.3083	5.7658	52.303	79.572	35.0	87.5
Total	15	67.833	20.1527	5.2034	56.673	78.994	25.0	90.0

Post Hoc Tests

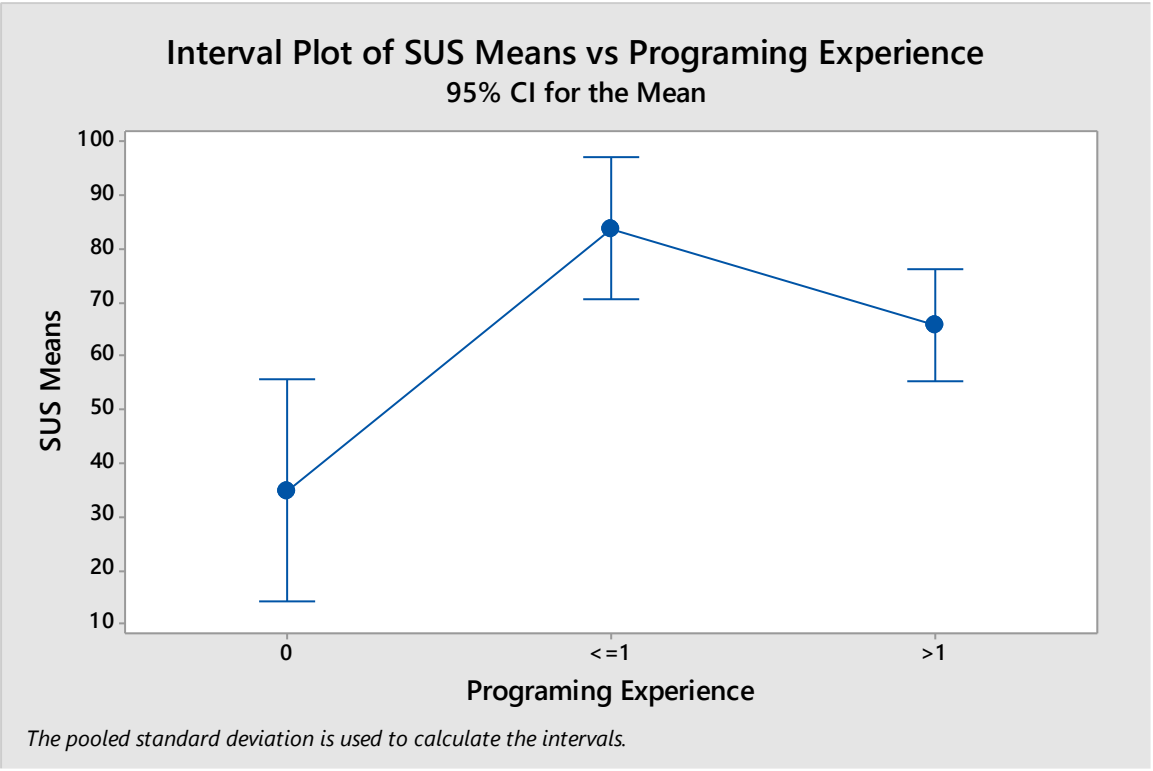
Multiple Comparisons

Dependent Variable: SUS_Total

LSD

(I) Pro_Exp	(J) Pro_Exp	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
No Experience	Year or less than 1 year	-49.0000*	11.3135	.001	-73.650	-24.350
	More than 1 year	-30.9375*	10.6903	.013	-54.230	-7.645
Year or less than 1 year	No Experience	49.0000*	11.3135	.001	24.350	73.650
	More than 1 year	18.0625*	7.7089	.037	1.266	34.859
More than 1 year	No Experience	30.9375*	10.6903	.013	7.645	54.230
	Year or less than 1 year	-18.0625*	7.7089	.037	-34.859	-1.266

*. The mean difference is significant at the 0.05 level.



Occupation

ANOVA

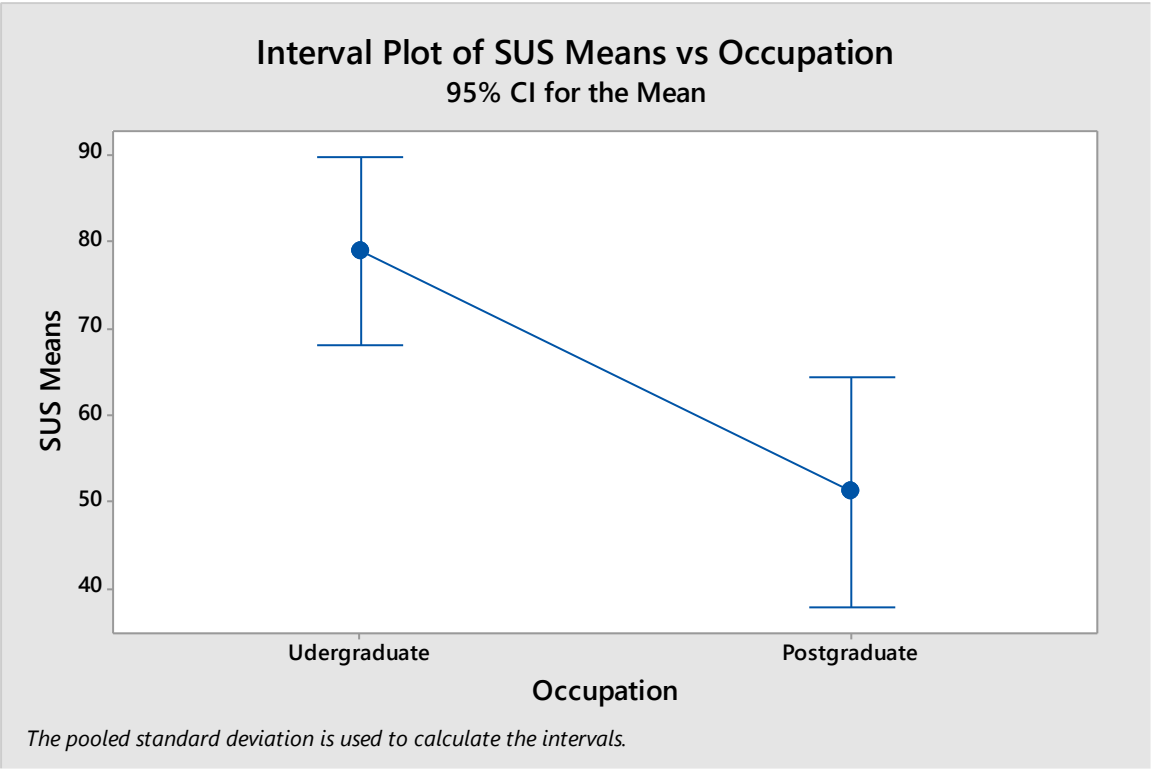
SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2750.069	1	2750.069	12.178	.004
Within Groups	2935.764	13	225.828		
Total	5685.833	14			

Descriptives

SUS_Total

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
1	9	78.889	11.1181	3.7060	70.343	87.435	55.0	90.0
2	6	51.250	19.7326	8.0558	30.542	71.958	25.0	75.0
Total	15	67.833	20.1527	5.2034	56.673	78.994	25.0	90.0



Means Plots

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet20.
DATASET ACTIVATE DataSet21.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1.458	1	1.458	.002	.962
Within Groups	7884.375	13	606.490		
Total	7885.833	14			

```
ONEWAY SUS_Total BY Age
```

```
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4117.976	3	1372.659	4.007	.037
Within Groups	3767.857	11	342.532		
Total	7885.833	14			

Descriptives

SUS_Total

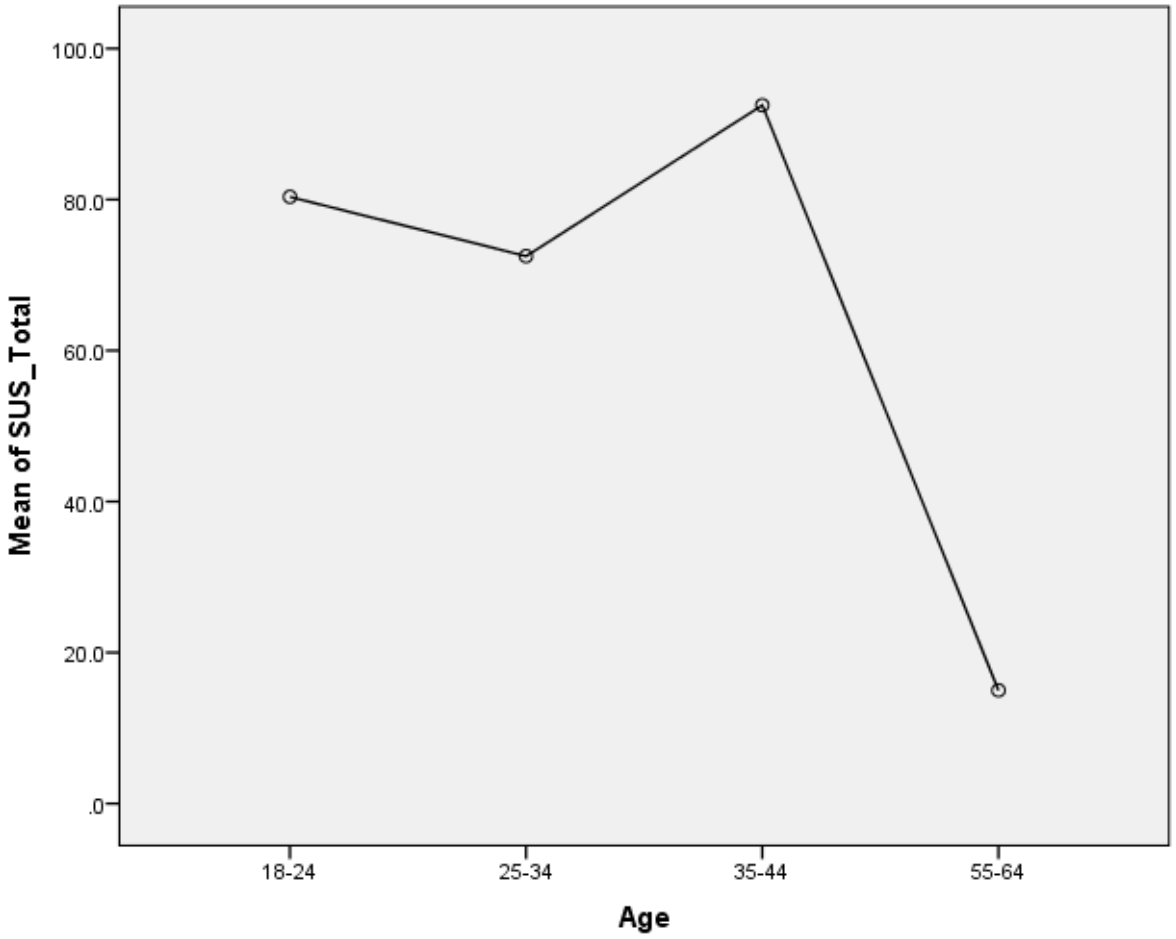
	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
18-24	7	80.357	16.3572	6.1825	65.229	95.485	52.5	95.0
25-34	6	72.500	20.7966	8.4902	50.675	94.325	47.5	97.5
35-44	1	92.500	.	.	.	.	92.5	92.5
55-64	1	15.000	.	.	.	.	15.0	15.0
Total	15	73.667	23.7334	6.1279	60.524	86.810	15.0	97.5

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4117.976	3	1372.659	4.007	.037
Within Groups	3767.857	11	342.532		
Total	7885.833	14			

Means Plots



Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	116.786	2	58.393	.090	.914
Within Groups	7769.048	12	647.421		
Total	7885.833	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	46.944	1	46.944	.078	.785
Within Groups	7838.889	13	602.991		
Total	7885.833	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet21.


DATASET CLOSE DataSet1.
ONEWAY SUS_Total BY Occupation
  /STATISTICS DESCRIPTIVES
  /PLOT MEANS
  /MISSING ANALYSIS
  /POSTHOC=LSD ALPHA(0.05).
```

SUS Web E overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		50.167
Median		50.000
Mode		37.5
Minimum		22.5
Maximum		85.0
Sum		752.5

a. App_code = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	6.7	6.7	6.7
	25.0	1	6.7	6.7	13.3
	30.0	1	6.7	6.7	20.0
	37.5	2	13.3	13.3	33.3
	40.0	1	6.7	6.7	40.0
	45.0	1	6.7	6.7	46.7
	50.0	1	6.7	6.7	53.3
	52.5	1	6.7	6.7	60.0

	55.0	1	6.7	6.7	66.7
	57.5	1	6.7	6.7	73.3
	65.0	1	6.7	6.7	80.0
	67.5	1	6.7	6.7	86.7
	82.5	1	6.7	6.7	93.3
	85.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

a. App_code = 1

SUS Desktop F overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		53.333
Median		60.000
Mode		62.5[b]
Minimum		10.0
Maximum		100.0
Sum		800.0

a. App_code = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	10.0	1	6.7	6.7	6.7
	17.5	1	6.7	6.7	13.3
	30.0	1	6.7	6.7	20.0
	37.5	1	6.7	6.7	26.7
	42.5	1	6.7	6.7	33.3
	45.0	1	6.7	6.7	40.0
	52.5	1	6.7	6.7	46.7
	60.0	1	6.7	6.7	53.3
	62.5	2	13.3	13.3	66.7
	65.0	2	13.3	13.3	80.0

	75.0	2	13.3	13.3	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

a. App_code = 2

```
SPLIT FILE OFF.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSH
V3\H_Web_V3.sav'.
DATASET NAME DataSet7 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web E and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		49.722
Median		45.000
Mode		25.0[b]
Minimum		25.0
Maximum		82.5
Sum		447.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	11.1	11.1	11.1
	30.0	1	11.1	11.1	22.2
	37.5	1	11.1	11.1	33.3
	40.0	1	11.1	11.1	44.4
	45.0	1	11.1	11.1	55.6

	55.0	1	11.1	11.1	66.7
	65.0	1	11.1	11.1	77.8
	67.5	1	11.1	11.1	88.9
	82.5	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		51.000
Median		52.500
Mode		22.5[b]
Minimum		22.5
Maximum		85.0
Sum		255.0

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	20.0	20.0	20.0
	37.5	1	20.0	20.0	40.0
	52.5	1	20.0	20.0	60.0
	57.5	1	20.0	20.0	80.0
	85.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		50.000
Median		50.000
Mode		50.0
Minimum		50.0
Maximum		50.0
Sum		50.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	50.0	1	100.0	100.0	100.0

a. Age = 35-44

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web E and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		46.429
Median		45.000
Mode		22.5[b]
Minimum		22.5

Maximum	85.0
Sum	325.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	14.3	14.3	14.3
	30.0	1	14.3	14.3	28.6
	37.5	1	14.3	14.3	42.9
	45.0	1	14.3	14.3	57.1
	50.0	1	14.3	14.3	71.4
	55.0	1	14.3	14.3	85.7
	85.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		59.375
Median		58.750
Mode		37.5[b]
Minimum		37.5
Maximum		82.5
Sum		237.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	25.0	25.0	25.0
	52.5	1	25.0	25.0	50.0
	65.0	1	25.0	25.0	75.0
	82.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		47.500
Median		48.750
Mode		25.0[b]
Minimum		25.0
Maximum		67.5
Sum		190.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	25.0	25.0	25.0
	40.0	1	25.0	25.0	50.0
	57.5	1	25.0	25.0	75.0
	67.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Gender.
```

```
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web E and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		54.750
Median		53.750
Mode		37.5
Minimum		25.0
Maximum		85.0
Sum		547.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	10.0	10.0	10.0
	37.5	2	20.0	20.0	30.0
	40.0	1	10.0	10.0	40.0
	50.0	1	10.0	10.0	50.0
	57.5	1	10.0	10.0	60.0
	65.0	1	10.0	10.0	70.0
	67.5	1	10.0	10.0	80.0
	82.5	1	10.0	10.0	90.0
	85.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		41.000
Median		45.000
Mode		22.5[b]
Minimum		22.5
Maximum		55.0
Sum		205.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	20.0	20.0	20.0
	30.0	1	20.0	20.0	40.0
	45.0	1	20.0	20.0	60.0
	52.5	1	20.0	20.0	80.0
	55.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Gender = Female

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSH
V3\H_Desktop_V3.sav'.
DATASET NAME DataSet8 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web E and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		49.643
Median		40.000
Mode		25.0[b]
Std. Deviation		21.8627
Minimum		25.0
Maximum		82.5
Sum		347.5

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	14.3	14.3	14.3
	30.0	1	14.3	14.3	28.6
	37.5	1	14.3	14.3	42.9
	40.0	1	14.3	14.3	57.1
	65.0	1	14.3	14.3	71.4
	67.5	1	14.3	14.3	85.7
	82.5	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		51.250
Median		52.500

Mode	22.5[b]
Std. Deviation	21.0208
Minimum	22.5
Maximum	85.0
Sum	307.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	22.5	1	16.7	16.7	16.7
	37.5	1	16.7	16.7	33.3
	50.0	1	16.7	16.7	50.0
	55.0	1	16.7	16.7	66.7
	57.5	1	16.7	16.7	83.3
	85.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		52.500
Median		52.500
Mode		52.5
Minimum		52.5
Maximum		52.5
Sum		52.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	100.0	100.0	100.0

a. Occupation = 4

Occupation = other occupations

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		45.000
Median		45.000
Mode		45.0
Minimum		45.0
Maximum		45.0
Sum		45.0

a. Occupation = 5

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	100.0	100.0	100.0

a. Occupation = 5

SUS Desktop E and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0

Mean	43.611
Median	45.000
Mode	10.0[b]
Minimum	10.0
Maximum	65.0
Sum	392.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	10.0	1	11.1	11.1	11.1
	17.5	1	11.1	11.1	22.2
	37.5	1	11.1	11.1	33.3
	42.5	1	11.1	11.1	44.4
	45.0	1	11.1	11.1	55.6
	52.5	1	11.1	11.1	66.7
	60.0	1	11.1	11.1	77.8
	62.5	1	11.1	11.1	88.9
	65.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		66.500
Median		65.000
Mode		30.0[b]
Minimum		30.0
Maximum		100.0
Sum		332.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	20.0	20.0	20.0
	62.5	1	20.0	20.0	40.0
	65.0	1	20.0	20.0	60.0
	75.0	1	20.0	20.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		75.000
Median		75.000
Mode		75.0
Minimum		75.0
Maximum		75.0
Sum		75.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	1	100.0	100.0	100.0

a. Age = 35-44

```
DATASET CLOSE DataSet7.
SORT CASES  BY Pro_Exp.
```

```
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop E and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		67.500
Median		65.000
Mode		75.0
Minimum		42.5
Maximum		100.0
Sum		472.5

a. Pro_Exp = No Experience

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	14.3	14.3	14.3
	52.5	1	14.3	14.3	28.6
	62.5	1	14.3	14.3	42.9
	65.0	1	14.3	14.3	57.1
	75.0	2	28.6	28.6	85.7
	100.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	4

	Missing	0
Mean		56.250
Median		61.250
Mode		37.5[b]
Minimum		37.5
Maximum		65.0
Sum		225.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	25.0	25.0	25.0
	60.0	1	25.0	25.0	50.0
	62.5	1	25.0	25.0	75.0
	65.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		25.625
Median		23.750
Mode		10.0[b]
Minimum		10.0
Maximum		45.0
Sum		102.5

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	10.0	1	25.0	25.0	25.0
	17.5	1	25.0	25.0	50.0
	30.0	1	25.0	25.0	75.0
	45.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop E and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		39.286
Median		42.500
Mode		10.0[b]
Minimum		10.0
Maximum		62.5
Sum		275.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	10.0	1	14.3	14.3	14.3
	17.5	1	14.3	14.3	28.6
	37.5	1	14.3	14.3	42.9
	42.5	1	14.3	14.3	57.1
	45.0	1	14.3	14.3	71.4
	60.0	1	14.3	14.3	85.7
	62.5	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		67.917
Median		70.000
Mode		75.0
Minimum		30.0
Maximum		100.0
Sum		407.5

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	16.7	16.7	16.7
	62.5	1	16.7	16.7	33.3
	65.0	1	16.7	16.7	50.0

	75.0	2	33.3	33.3	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		65.000
Median		65.000
Mode		65.0
Minimum		65.0
Maximum		65.0
Sum		65.0

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	65.0	1	100.0	100.0	100.0

a. Occupation = 4

Occupation = other occupations

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		52.500

Median	52.500
Mode	52.5
Minimum	52.5
Maximum	52.5
Sum	52.5

a. Occupation = 5

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	100.0	100.0	100.0

a. Occupation = 5

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSH
V3\H_Web_V3.sav'.
DATASET NAME DataSet16 WINDOW=FRONT.
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.


GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSH
V3\H_Desktop_V3.sav'.
DATASET NAME DataSet17 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop E and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		50.000
Median		52.500

Mode	62.5
Std. Deviation	27.4368
Minimum	10.0
Maximum	100.0
Sum	500.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	10.0	1	10.0	10.0	10.0
	17.5	1	10.0	10.0	20.0
	30.0	1	10.0	10.0	30.0
	37.5	1	10.0	10.0	40.0
	45.0	1	10.0	10.0	50.0
	60.0	1	10.0	10.0	60.0
	62.5	2	20.0	20.0	80.0
	75.0	1	10.0	10.0	90.0
	100.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		60.000
Median		65.000
Mode		65.0
Std. Deviation		12.6244
Minimum		42.5
Maximum		75.0
Sum		300.0

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	20.0	20.0	20.0
	52.5	1	20.0	20.0	40.0
	65.0	2	40.0	40.0	80.0
	75.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Gender = Female

```
DATASET ACTIVATE DataSet16.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for E Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	630.208	1	630.208	1.825	.200
Within Groups	4488.125	13	345.240		
Total	5118.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	5.278	2	2.639	.006	.994
Within Groups	5113.056	12	426.088		
Total	5118.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	465.432	2	232.716	.600	.564
Within Groups	4652.902	12	387.742		
Total	5118.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	41.101	3	13.700	.030	.993
Within Groups	5077.232	11	461.567		
Total	5118.333	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet16.
DATASET ACTIVATE DataSet17.
SPLIT FILE OFF.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gende

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	333.333	1	333.333	.585	.458
Within Groups	7412.500	13	570.192		
Total	7745.833	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2186.944	2	1093.472	2.360	.137
Within Groups	5558.889	12	463.241		
Total	7745.833	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

Descriptives

SUS_Total

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
No Experience	7	67.500	18.4842	6.9864	50.405	84.595	42.5	100.0
Year or less than 1 year	4	56.250	12.6656	6.3328	36.096	76.404	37.5	65.0
More than 1 year	4	25.625	15.3263	7.6632	1.237	50.013	10.0	45.0
Total	15	53.333	23.5218	6.0733	40.307	66.359	10.0	100.0

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4509.896	2	2254.948	8.362	.005
Within Groups	3235.938	12	269.661		

Total	7745.833	14			

Post Hoc Tests

Multiple Comparisons

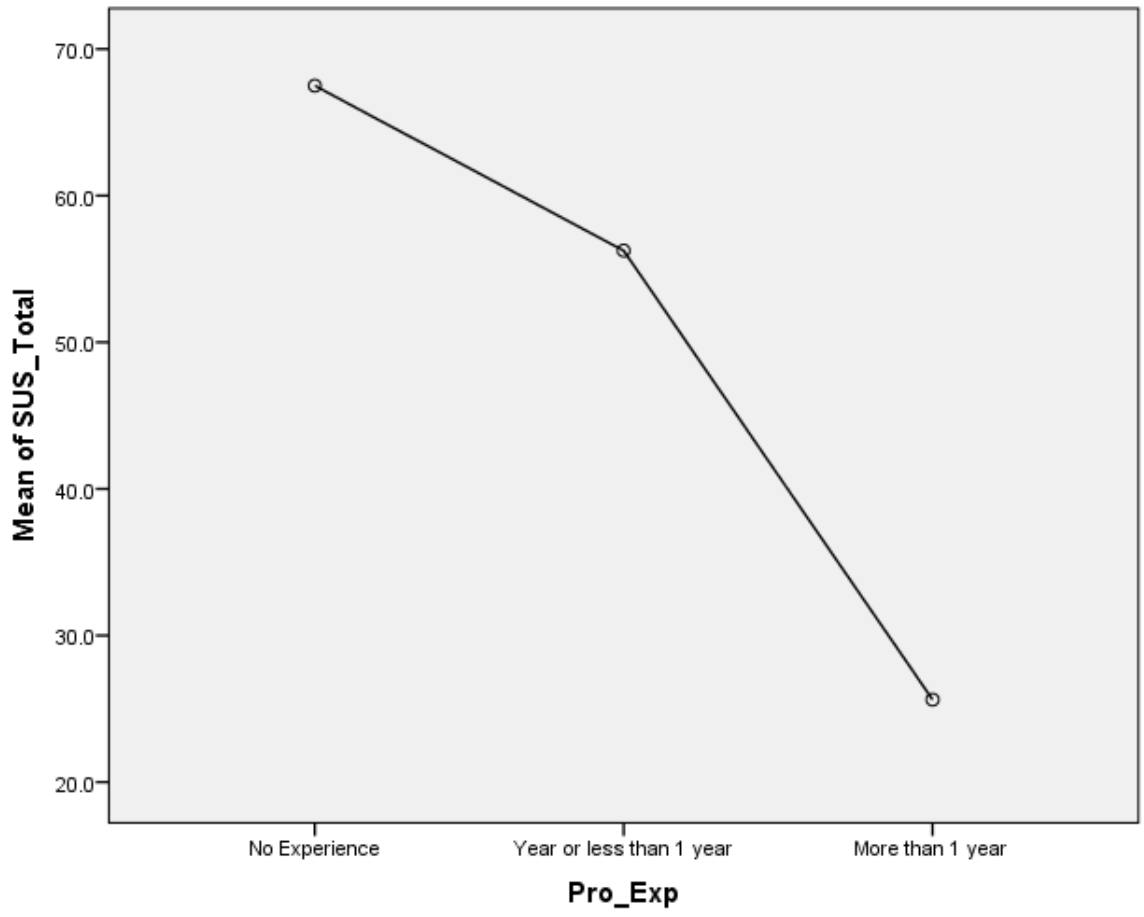
Dependent Variable: SUS_Total

Tukey HSD

(I) Pro_Exp	(J) Pro_Exp	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
No Experience	Year or less than 1 year	11.2500	10.2926	.536	-16.209	38.709
	More than 1 year	41.8750*	10.2926	.004	14.416	69.334
Year or less than 1 year	No Experience	-11.2500	10.2926	.536	-38.709	16.209
	More than 1 year	30.6250	11.6117	.053	-.353	61.603
More than 1 year	No Experience	-41.8750*	10.2926	.004	-69.334	-14.416
	Year or less than 1 year	-30.6250	11.6117	.053	-61.603	.353

*. The mean difference is significant at the 0.05 level.

Means Plots



```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	2794.196	3	931.399	2.069	.163
Within Groups	4951.637	11	450.149		
Total	7745.833	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet17.
```

SUS Web I overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		70.333
Median		77.500
Mode		87.5
Minimum		35.0
Maximum		100.0
Sum		1055.0

a. App_code = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	6.7	6.7	6.7
	42.5	1	6.7	6.7	13.3

	45.0	1	6.7	6.7	20.0
	47.5	1	6.7	6.7	26.7
	55.0	1	6.7	6.7	33.3
	60.0	1	6.7	6.7	40.0
	72.5	1	6.7	6.7	46.7
	77.5	1	6.7	6.7	53.3
	80.0	1	6.7	6.7	60.0
	82.5	1	6.7	6.7	66.7
	87.5	2	13.3	13.3	80.0
	90.0	1	6.7	6.7	86.7
	92.5	1	6.7	6.7	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

a. App_code = 1

SUS Desktop I overall

Statistics[a]

SUS_Total

N	Valid	15
	Missing	0
Mean		68.167
Median		67.500
Mode		55.0[b]
Minimum		35.0
Maximum		97.5
Sum		1022.5

a. App_code = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	6.7	6.7	6.7
	37.5	1	6.7	6.7	13.3
	50.0	1	6.7	6.7	20.0

55.0	2	13.3	13.3	33.3
65.0	2	13.3	13.3	46.7
67.5	2	13.3	13.3	60.0
72.5	1	6.7	6.7	66.7
77.5	1	6.7	6.7	73.3
87.5	1	6.7	6.7	80.0
95.0	2	13.3	13.3	93.3
97.5	1	6.7	6.7	100.0
Total	15	100.0	100.0	

a. App_code = 2

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSJ
V3\J_Web_V3.sav'.
DATASET NAME DataSet10 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web I and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		67.885
Median		72.500
Mode		87.5
Std. Deviation		21.5263
Minimum		35.0
Maximum		100.0
Sum		882.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	7.7	7.7	7.7
	42.5	1	7.7	7.7	15.4
	45.0	1	7.7	7.7	23.1
	47.5	1	7.7	7.7	30.8
	55.0	1	7.7	7.7	38.5
	60.0	1	7.7	7.7	46.2
	72.5	1	7.7	7.7	53.8
	77.5	1	7.7	7.7	61.5
	80.0	1	7.7	7.7	69.2
	87.5	2	15.4	15.4	84.6
	92.5	1	7.7	7.7	92.3
	100.0	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		86.250
Median		86.250
Mode		82.5[b]
Std. Deviation		5.3033
Minimum		82.5
Maximum		90.0
Sum		172.5

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

	82.5	1	50.0	50.0	50.0
Valid	90.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web I and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		71.136
Median		77.500
Mode		87.5
Std. Deviation		20.0737
Minimum		35.0
Maximum		92.5
Sum		782.5

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
	35.0	1	9.1	9.1	9.1
	42.5	1	9.1	9.1	18.2
Valid	55.0	1	9.1	9.1	27.3
	60.0	1	9.1	9.1	36.4
	72.5	1	9.1	9.1	45.5

	77.5	1	9.1	9.1	54.5
	82.5	1	9.1	9.1	63.6
	87.5	2	18.2	18.2	81.8
	90.0	1	9.1	9.1	90.9
	92.5	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		57.500
Median		47.500
Mode		45.0[b]
Std. Deviation		19.5256
Minimum		45.0
Maximum		80.0
Sum		172.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	33.3	33.3	33.3
	47.5	1	33.3	33.3	66.7
	80.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		100.000
Median		100.000
Mode		100.0
Minimum		100.0
Maximum		100.0
Sum		100.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	100.0	1	100.0	100.0	100.0

a. Age = 35-44

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web I and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		64.000
Median		55.000
Mode		45.0[b]
Std. Deviation		20.8117
Minimum		45.0
Maximum		90.0
Sum		320.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	20.0	20.0	20.0
	47.5	1	20.0	20.0	40.0
	55.0	1	20.0	20.0	60.0
	82.5	1	20.0	20.0	80.0
	90.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		86.667
Median		87.500
Mode		72.5[b]
Std. Deviation		13.7689
Minimum		72.5
Maximum		100.0
Sum		260.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

Valid	72.5	1	33.3	33.3	33.3
	87.5	1	33.3	33.3	66.7
	100.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		67.857
Median		77.500
Mode		35.0[b]
Std. Deviation		22.4271
Minimum		35.0
Maximum		92.5
Sum		475.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	14.3	14.3	14.3
	42.5	1	14.3	14.3	28.6
	60.0	1	14.3	14.3	42.9
	77.5	1	14.3	14.3	57.1
	80.0	1	14.3	14.3	71.4
	87.5	1	14.3	14.3	85.7
	92.5	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
```

```
  /ORDER=ANALYSIS.
```

SUS Web I and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		71.136
Median		77.500
Mode		87.5
Std. Deviation		20.0737
Minimum		35.0
Maximum		92.5
Sum		782.5

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	9.1	9.1	9.1
	42.5	1	9.1	9.1	18.2
	55.0	1	9.1	9.1	27.3
	60.0	1	9.1	9.1	36.4
	72.5	1	9.1	9.1	45.5
	77.5	1	9.1	9.1	54.5
	82.5	1	9.1	9.1	63.6
	87.5	2	18.2	18.2	81.8
	90.0	1	9.1	9.1	90.9
	92.5	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		68.125
Median		63.750
Mode		45.0[b]
Std. Deviation		26.5656
Minimum		45.0
Maximum		100.0
Sum		272.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	45.0	1	25.0	25.0	25.0
	47.5	1	25.0	25.0	50.0
	80.0	1	25.0	25.0	75.0
	100.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Occupation = 2

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSJ
V3\J_Desktop_V3.sav'.
DATASET NAME DataSet11 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop I and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		71.731
Median		67.500
Mode		65.0[b]
Std. Deviation		18.4387
Minimum		37.5
Maximum		97.5
Sum		932.5

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	7.7	7.7	7.7
	50.0	1	7.7	7.7	15.4
	55.0	1	7.7	7.7	23.1
	65.0	2	15.4	15.4	38.5
	67.5	2	15.4	15.4	53.8
	72.5	1	7.7	7.7	61.5
	77.5	1	7.7	7.7	69.2
	87.5	1	7.7	7.7	76.9
	95.0	2	15.4	15.4	92.3
	97.5	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		45.000
Median		45.000
Mode		35.0[b]
Std. Deviation		14.1421
Minimum		35.0
Maximum		55.0
Sum		90.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	50.0	50.0	50.0
	55.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop I and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	11

	Missing	0
Mean		62.955
Median		65.000
Mode		55.0[b]
Std. Deviation		19.0006
Minimum		35.0
Maximum		95.0
Sum		692.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	9.1	9.1	9.1
	37.5	1	9.1	9.1	18.2
	50.0	1	9.1	9.1	27.3
	55.0	2	18.2	18.2	45.5
	65.0	1	9.1	9.1	54.5
	67.5	2	18.2	18.2	72.7
	77.5	1	9.1	9.1	81.8
	87.5	1	9.1	9.1	90.9
	95.0	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		77.500
Median		72.500
Mode		65.0[b]
Std. Deviation		15.6125

Minimum	65.0
Maximum	95.0
Sum	232.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	65.0	1	33.3	33.3	33.3
	72.5	1	33.3	33.3	66.7
	95.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		97.500
Median		97.500
Mode		97.5
Minimum		97.5
Maximum		97.5
Sum		97.5

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	97.5	1	100.0	100.0	100.0

a. Age = 35-44

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop I and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		56.500
Median		55.000
Mode		55.0
Std. Deviation		14.0979
Minimum		35.0
Maximum		72.5
Sum		282.5

a. Pro_Exp = No Experience

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	20.0	20.0	20.0
	55.0	2	40.0	40.0	60.0
	65.0	1	20.0	20.0	80.0
	72.5	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		76.667
Median		67.500
Mode		65.0[b]
Std. Deviation		18.0854
Minimum		65.0
Maximum		97.5
Sum		230.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	65.0	1	33.3	33.3	33.3
	67.5	1	33.3	33.3	66.7
	97.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		72.857
Median		77.500
Mode		95.0
Std. Deviation		22.4271
Minimum		37.5

Maximum	95.0
Sum	510.0

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	14.3	14.3	14.3
	50.0	1	14.3	14.3	28.6
	67.5	1	14.3	14.3	42.9
	77.5	1	14.3	14.3	57.1
	87.5	1	14.3	14.3	71.4
	95.0	2	28.6	28.6	100.0
	Total	7	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop I and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		62.955
Median		65.000
Mode		55.0[b]
Std. Deviation		19.0006
Minimum		35.0
Maximum		95.0
Sum		692.5

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	9.1	9.1	9.1
	37.5	1	9.1	9.1	18.2
	50.0	1	9.1	9.1	27.3
	55.0	2	18.2	18.2	45.5
	65.0	1	9.1	9.1	54.5
	67.5	2	18.2	18.2	72.7
	77.5	1	9.1	9.1	81.8
	87.5	1	9.1	9.1	90.9
	95.0	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		82.500
Median		83.750
Mode		65.0[b]
Std. Deviation		16.2019
Minimum		65.0
Maximum		97.5
Sum		330.0

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	65.0	1	25.0	25.0	25.0
	72.5	1	25.0	25.0	50.0
	95.0	1	25.0	25.0	75.0
	97.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Occupation = 2

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSJ
V3\J_Web_V3.sav'.
DATASET NAME DataSet24 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for I Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	584.631	1	584.631	1.360	.264
Within Groups	5588.702	13	429.900		
Total	6173.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.

Between Groups	1381.288	2	690.644	1.729	.219
Within Groups	4792.045	12	399.337		
Total	6173.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1043.810	2	521.905	1.221	.329
Within Groups	5129.524	12	427.460		
Total	6173.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	26.600	1	26.600	.056	.816
Within Groups	6146.733	13	472.826		
Total	6173.333	14			

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSJ
V3\J_Desktop_V3.sav'.
DATASET NAME DataSet25 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1238.526	1	1238.526	3.762	.074
Within Groups	4279.808	13	329.216		
Total	5518.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1420.606	2	710.303	2.080	.168
Within Groups	4097.727	12	341.477		
Total	5518.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1051.310	2	525.655	1.412	.281
Within Groups	4467.024	12	372.252		
Total	5518.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1120.606	1	1120.606	3.313	.092
Within Groups	4397.727	13	338.287		
Total	5518.333	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet25.
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet24.
```

SUS Web H overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		37.000
Median		30.000
Mode		12.5
Std. Deviation		26.4271
Minimum		5.0
Maximum		87.5
Sum		555.0

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	5.0	1	6.7	6.7	6.7
	10.0	1	6.7	6.7	13.3
	12.5	2	13.3	13.3	26.7
	15.0	1	6.7	6.7	33.3
	17.5	1	6.7	6.7	40.0
	27.5	1	6.7	6.7	46.7
	30.0	1	6.7	6.7	53.3
	37.5	1	6.7	6.7	60.0
	42.5	1	6.7	6.7	66.7
	57.5	1	6.7	6.7	73.3
	60.0	1	6.7	6.7	80.0
	62.5	1	6.7	6.7	86.7

	77.5	1	6.7	6.7	93.3
	87.5	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSM
V3\M_Desktop_V3.sav'.
DATASET NAME DataSet26 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop H overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		75.500
Median		77.500
Mode		85.0
Std. Deviation		13.7321
Minimum		52.5
Maximum		100.0
Sum		1132.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	2	13.3	13.3	13.3
	62.5	1	6.7	6.7	20.0
	65.0	1	6.7	6.7	26.7
	70.0	2	13.3	13.3	40.0
	72.5	1	6.7	6.7	46.7
	77.5	1	6.7	6.7	53.3
	80.0	1	6.7	6.7	60.0
	82.5	1	6.7	6.7	66.7
	85.0	3	20.0	20.0	86.7

	92.5	1	6.7	6.7	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
DATASET ACTIVATE DataSet24.
DATASET CLOSE DataSet26.
DATASET ACTIVATE DataSet25.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web H and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		35.000
Median		30.000
Mode		5.0[b]
Std. Deviation		25.4214
Minimum		5.0
Maximum		87.5
Sum		385.0

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	5.0	1	9.1	9.1	9.1
	10.0	1	9.1	9.1	18.2
	12.5	1	9.1	9.1	27.3
	15.0	1	9.1	9.1	36.4

	27.5	1	9.1	9.1	45.5
	30.0	1	9.1	9.1	54.5
	37.5	1	9.1	9.1	63.6
	42.5	1	9.1	9.1	72.7
	57.5	1	9.1	9.1	81.8
	60.0	1	9.1	9.1	90.9
	87.5	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		42.500
Median		40.000
Mode		12.5[b]
Std. Deviation		32.4037
Minimum		12.5
Maximum		77.5
Sum		170.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	12.5	1	25.0	25.0	25.0
	17.5	1	25.0	25.0	50.0
	62.5	1	25.0	25.0	75.0
	77.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web H and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		34.773
Median		30.000
Mode		5.0[b]
Std. Deviation		23.8079
Minimum		5.0
Maximum		77.5
Sum		382.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	5.0	1	9.1	9.1	9.1
	10.0	1	9.1	9.1	18.2
	12.5	1	9.1	9.1	27.3
	17.5	1	9.1	9.1	36.4
	27.5	1	9.1	9.1	45.5
	30.0	1	9.1	9.1	54.5
	37.5	1	9.1	9.1	63.6
	42.5	1	9.1	9.1	72.7

	60.0	1	9.1	9.1	81.8
	62.5	1	9.1	9.1	90.9
	77.5	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		43.125
Median		36.250
Mode		12.5[b]
Std. Deviation		36.0772
Minimum		12.5
Maximum		87.5
Sum		172.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	12.5	1	25.0	25.0	25.0
	15.0	1	25.0	25.0	50.0
	57.5	1	25.0	25.0	75.0
	87.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Age = 25-34

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
```

```
  /ORDER=ANALYSIS.
```

SUS Web H and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		30.000
Median		15.000
Mode		12.5[b]
Std. Deviation		28.1736
Minimum		12.5
Maximum		62.5
Sum		90.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	12.5	1	33.3	33.3	33.3
	15.0	1	33.3	33.3	66.7
	62.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		20.000
Median		17.500
Mode		12.5[b]
Std. Deviation		9.0139
Minimum		12.5
Maximum		30.0
Sum		60.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	12.5	1	33.3	33.3	33.3
	17.5	1	33.3	33.3	66.7
	30.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		45.000
Median		42.500
Mode		5.0[b]
Std. Deviation		28.3670
Minimum		5.0
Maximum		87.5
Sum		405.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	5.0	1	11.1	11.1	11.1
	10.0	1	11.1	11.1	22.2
	27.5	1	11.1	11.1	33.3
	37.5	1	11.1	11.1	44.4
	42.5	1	11.1	11.1	55.6
	57.5	1	11.1	11.1	66.7
	60.0	1	11.1	11.1	77.8
	77.5	1	11.1	11.1	88.9
	87.5	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web H and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		31.944
Median		30.000
Mode		5.0[b]
Std. Deviation		20.8333
Minimum		5.0
Maximum		62.5
Sum		287.5

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	5.0	1	11.1	11.1	11.1
	10.0	1	11.1	11.1	22.2
	12.5	1	11.1	11.1	33.3
	27.5	1	11.1	11.1	44.4
	30.0	1	11.1	11.1	55.6
	37.5	1	11.1	11.1	66.7
	42.5	1	11.1	11.1	77.8
	60.0	1	11.1	11.1	88.9
	62.5	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		44.583
Median		37.500
Mode		12.5[b]
Std. Deviation		33.8532
Minimum		12.5
Maximum		87.5
Sum		267.5

a. Occupation = 2

b. Multiple modes exist. The

smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	12.5	1	16.7	16.7	16.7
	15.0	1	16.7	16.7	33.3
	17.5	1	16.7	16.7	50.0
	57.5	1	16.7	16.7	66.7
	77.5	1	16.7	16.7	83.3
	87.5	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

```
DATASET ACTIVATE DataSet24.
DATASET CLOSE DataSet25.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSM
V3\M_Desktop_V3.sav'.
DATASET NAME DataSet27 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop H and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0

Mean	78.542
Median	78.750
Mode	70.0[b]
Std. Deviation	11.2036
Minimum	62.5
Maximum	100.0
Sum	942.5

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	62.5	1	8.3	8.3	8.3
	65.0	1	8.3	8.3	16.7
	70.0	2	16.7	16.7	33.3
	72.5	1	8.3	8.3	41.7
	77.5	1	8.3	8.3	50.0
	80.0	1	8.3	8.3	58.3
	82.5	1	8.3	8.3	66.7
	85.0	2	16.7	16.7	83.3
	92.5	1	8.3	8.3	91.7
	100.0	1	8.3	8.3	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		63.333
Median		52.500
Mode		52.5

Std. Deviation	18.7639
Minimum	52.5
Maximum	85.0
Sum	190.0

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	2	66.7	66.7	66.7
	85.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop H and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	11
	Missing	0
Mean		72.727
Median		72.500
Mode		52.5[b]
Std. Deviation		13.5303
Minimum		52.5
Maximum		92.5
Sum		800.0

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	2	18.2	18.2	18.2
	62.5	1	9.1	9.1	27.3
	65.0	1	9.1	9.1	36.4
	70.0	1	9.1	9.1	45.5
	72.5	1	9.1	9.1	54.5
	80.0	1	9.1	9.1	63.6
	82.5	1	9.1	9.1	72.7
	85.0	2	18.2	18.2	90.9
	92.5	1	9.1	9.1	100.0
	Total	11	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		83.125
Median		81.250
Mode		70.0[b]
Std. Deviation		12.8087
Minimum		70.0
Maximum		100.0
Sum		332.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	25.0	25.0	25.0
	77.5	1	25.0	25.0	50.0
	85.0	1	25.0	25.0	75.0
	100.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Age = 25-34

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop H and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		69.167
Median		70.000
Mode		52.5[b]
Std. Deviation		16.2660
Minimum		52.5
Maximum		85.0
Sum		207.5

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	33.3	33.3	33.3
	70.0	1	33.3	33.3	66.7

	85.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		71.667
Median		77.500
Mode		52.5[b]
Std. Deviation		17.0171
Minimum		52.5
Maximum		85.0
Sum		215.0

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	33.3	33.3	33.3
	77.5	1	33.3	33.3	66.7
	85.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0

Mean	78.889
Median	80.000
Mode	62.5[b]
Std. Deviation	12.5693
Minimum	62.5
Maximum	100.0
Sum	710.0

a. Pro_Exp = More than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	62.5	1	11.1	11.1	11.1
	65.0	1	11.1	11.1	22.2
	70.0	1	11.1	11.1	33.3
	72.5	1	11.1	11.1	44.4
	80.0	1	11.1	11.1	55.6
	82.5	1	11.1	11.1	66.7
	85.0	1	11.1	11.1	77.8
	92.5	1	11.1	11.1	88.9
	100.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop H and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	10

	Missing	0
Mean		74.750
Median		76.250
Mode		85.0
Std. Deviation		12.3856
Minimum		52.5
Maximum		92.5
Sum		747.5

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	10.0	10.0	10.0
	62.5	1	10.0	10.0	20.0
	65.0	1	10.0	10.0	30.0
	70.0	1	10.0	10.0	40.0
	72.5	1	10.0	10.0	50.0
	80.0	1	10.0	10.0	60.0
	82.5	1	10.0	10.0	70.0
	85.0	2	20.0	20.0	90.0
	92.5	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		77.000
Median		77.500
Mode		52.5[b]
Std. Deviation		17.6246
Minimum		52.5

Maximum	100.0
Sum	385.0

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	20.0	20.0	20.0
	70.0	1	20.0	20.0	40.0
	77.5	1	20.0	20.0	60.0
	85.0	1	20.0	20.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Occupation = 2

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSM
V3\M_Web_V3.sav'.
DATASET NAME DataSet22 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for F Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	165.000	1	165.000	.223	.644
Within Groups	9612.500	13	739.423		
Total	9777.500	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	204.631	1	204.631	.278	.607
Within Groups	9572.869	13	736.375		
Total	9777.500	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1590.000	2	795.000	1.165	.345
Within Groups	8187.500	12	682.292		
Total	9777.500	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	575.069	1	575.069	.812	.384
Within Groups	9202.431	13	707.879		
Total	9777.500	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet22.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSM
V3\M_Desktop_V3.sav'.
DATASET NAME DataSet23 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	555.104	1	555.104	3.461	.086
Within Groups	2084.896	13	160.377		
Total	2640.000	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	317.131	1	317.131	1.775	.206
Within Groups	2322.869	13	178.682		
Total	2640.000	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	267.778	2	133.889	.677	.526
Within Groups	2372.222	12	197.685		
Total	2640.000	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	16.875	1	16.875	.084	.777
Within Groups	2623.125	13	201.779		
Total	2640.000	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet23.
```

SUS Web A overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		82.500
Median		82.500
Mode		75.0
Std. Deviation		8.5042
Minimum		70.0
Maximum		95.0
Sum		1237.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	6.7	6.7	6.7
	72.5	1	6.7	6.7	13.3
	75.0	4	26.7	26.7	40.0
	80.0	1	6.7	6.7	46.7
	82.5	1	6.7	6.7	53.3
	85.0	1	6.7	6.7	60.0

	87.5	2	13.3	13.3	73.3
	90.0	1	6.7	6.7	80.0
	92.5	1	6.7	6.7	86.7
	95.0	2	13.3	13.3	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCA
V3\DCA_Desktop_V3.sav'.
DATASET NAME DataSet30 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop A overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		77.667
Median		85.000
Mode		90.0
Std. Deviation		18.6956
Minimum		30.0
Maximum		100.0
Sum		1165.0

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	6.7	6.7	6.7
	52.5	1	6.7	6.7	13.3
	65.0	1	6.7	6.7	20.0
	67.5	2	13.3	13.3	33.3
	75.0	1	6.7	6.7	40.0
	77.5	1	6.7	6.7	46.7
	85.0	1	6.7	6.7	53.3

	90.0	5	33.3	33.3	86.7
	95.0	1	6.7	6.7	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

```
DATASET ACTIVATE DataSet24.
DATASET CLOSE DataSet30.
DATASET ACTIVATE DataSet29.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web A and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	14
	Missing	0
Mean		81.607
Median		81.250
Mode		75.0
Std. Deviation		8.0627
Minimum		70.0
Maximum		95.0
Sum		1142.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	7.1	7.1	7.1
	72.5	1	7.1	7.1	14.3
	75.0	4	28.6	28.6	42.9
	80.0	1	7.1	7.1	50.0

	82.5	1	7.1	7.1	57.1
	85.0	1	7.1	7.1	64.3
	87.5	2	14.3	14.3	78.6
	90.0	1	7.1	7.1	85.7
	92.5	1	7.1	7.1	92.9
	95.0	1	7.1	7.1	100.0
	Total	14	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		95.000
Median		95.000
Mode		95.0
Minimum		95.0
Maximum		95.0
Sum		95.0

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	95.0	1	100.0	100.0	100.0

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web A and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		84.500
Median		86.250
Mode		75.0
Std. Deviation		7.4349
Minimum		75.0
Maximum		95.0
Sum		845.0

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	3	30.0	30.0	30.0
	82.5	1	10.0	10.0	40.0
	85.0	1	10.0	10.0	50.0
	87.5	2	20.0	20.0	70.0
	90.0	1	10.0	10.0	80.0
	92.5	1	10.0	10.0	90.0
	95.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		78.500
Median		75.000
Mode		70.0[b]

Std. Deviation	9.9373
Minimum	70.0
Maximum	95.0
Sum	392.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	20.0	20.0	20.0
	72.5	1	20.0	20.0	40.0
	75.0	1	20.0	20.0	60.0
	80.0	1	20.0	20.0	80.0
	95.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web A and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		77.500
Median		77.500
Mode		75.0[b]

Std. Deviation	3.5355
Minimum	75.0
Maximum	80.0
Sum	155.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	1	50.0	50.0	50.0
	80.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		80.000
Median		78.750
Mode		75.0
Std. Deviation		6.1237
Minimum		75.0
Maximum		87.5
Sum		320.0

a. Pro_Exp = Year or less than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	2	50.0	50.0	50.0
	82.5	1	25.0	25.0	75.0
	87.5	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		84.722
Median		87.500
Mode		95.0
Std. Deviation		9.7983
Minimum		70.0
Maximum		95.0
Sum		762.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	11.1	11.1	11.1
	72.5	1	11.1	11.1	22.2
	75.0	1	11.1	11.1	33.3
	85.0	1	11.1	11.1	44.4
	87.5	1	11.1	11.1	55.6
	90.0	1	11.1	11.1	66.7

	92.5	1	11.1	11.1	77.8
	95.0	2	22.2	22.2	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web A and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		84.375
Median		86.250
Mode		75.0[b]
Std. Deviation		6.5124
Minimum		75.0
Maximum		92.5
Sum		675.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	2	25.0	25.0	25.0
	82.5	1	12.5	12.5	37.5

	85.0	1	12.5	12.5	50.0
	87.5	2	25.0	25.0	75.0
	90.0	1	12.5	12.5	87.5
	92.5	1	12.5	12.5	100.0
	Total	8	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		80.357
Median		75.000
Mode		75.0[b]
Std. Deviation		10.4511
Minimum		70.0
Maximum		95.0
Sum		562.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	14.3	14.3	14.3
	72.5	1	14.3	14.3	28.6
	75.0	2	28.6	28.6	57.1
	80.0	1	14.3	14.3	71.4
	95.0	2	28.6	28.6	100.0
	Total	7	100.0	100.0	

a. Occupation = 2

SUS desktop A and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	14
	Missing	0
Mean		76.071
Median		81.250
Mode		90.0
Std. Deviation		18.3113
Minimum		30.0
Maximum		95.0
Sum		1065.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	7.1	7.1	7.1
	52.5	1	7.1	7.1	14.3
	65.0	1	7.1	7.1	21.4
	67.5	2	14.3	14.3	35.7
	75.0	1	7.1	7.1	42.9
	77.5	1	7.1	7.1	50.0
	85.0	1	7.1	7.1	57.1
	90.0	5	35.7	35.7	92.9
	95.0	1	7.1	7.1	100.0
	Total	14	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		100.000
Median		100.000
Mode		100.0
Minimum		100.0
Maximum		100.0
Sum		100.0

a. Gender = Female

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	100.0	1	100.0	100.0	100.0

a. Gender = Female

SUS desktop A and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	10
	Missing	0
Mean		76.500
Median		87.500
Mode		90.0
Std. Deviation		22.2736
Minimum		30.0
Maximum		100.0
Sum		765.0

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	10.0	10.0	10.0
	52.5	1	10.0	10.0	20.0

	65.0	1	10.0	10.0	30.0
	67.5	1	10.0	10.0	40.0
	85.0	1	10.0	10.0	50.0
	90.0	3	30.0	30.0	80.0
	95.0	1	10.0	10.0	90.0
	100.0	1	10.0	10.0	100.0
	Total	10	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		80.000
Median		77.500
Mode		90.0
Std. Deviation		9.8425
Minimum		67.5
Maximum		90.0
Sum		400.0

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	67.5	1	20.0	20.0	20.0
	75.0	1	20.0	20.0	40.0
	77.5	1	20.0	20.0	60.0
	90.0	2	40.0	40.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

SUS desktop A and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		82.500
Median		82.500
Mode		75.0[b]
Std. Deviation		10.6066
Minimum		75.0
Maximum		90.0
Sum		165.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	75.0	1	50.0	50.0	50.0
	90.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		63.125
Median		66.250
Mode		30.0[b]
Std. Deviation		24.7803
Minimum		30.0
Maximum		90.0
Sum		252.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	25.0	25.0	25.0
	65.0	1	25.0	25.0	50.0
	67.5	1	25.0	25.0	75.0
	90.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		83.056
Median		90.000
Mode		90.0
Std. Deviation		14.9362
Minimum		52.5
Maximum		100.0
Sum		747.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	11.1	11.1	11.1
	67.5	1	11.1	11.1	22.2
	77.5	1	11.1	11.1	33.3
	85.0	1	11.1	11.1	44.4
	90.0	3	33.3	33.3	77.8
	95.0	1	11.1	11.1	88.9
	100.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

SUS desktop A and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		71.875
Median		76.250
Mode		90.0
Std. Deviation		22.5495
Minimum		30.0
Maximum		95.0
Sum		575.0

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	30.0	1	12.5	12.5	12.5
	52.5	1	12.5	12.5	25.0
	65.0	1	12.5	12.5	37.5
	67.5	1	12.5	12.5	50.0
	85.0	1	12.5	12.5	62.5
	90.0	2	25.0	25.0	87.5
	95.0	1	12.5	12.5	100.0
	Total	8	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		84.286
Median		90.000
Mode		90.0
Std. Deviation		11.2467
Minimum		67.5
Maximum		100.0
Sum		590.0

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	67.5	1	14.3	14.3	14.3
	75.0	1	14.3	14.3	28.6
	77.5	1	14.3	14.3	42.9
	90.0	3	42.9	42.9	85.7
	100.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Occupation = 2

ANOVA Testing for A Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	167.411	1	167.411	2.575	.133

Within Groups	845.089	13	65.007		
Total	1012.500	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	120.000	1	120.000	1.748	.209
Within Groups	892.500	13	68.654		
Total	1012.500	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Programming experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	119.444	2	59.722	.802	.471
Within Groups	893.056	12	74.421		
Total	1012.500	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	60.268	1	60.268	.823	.381
Within Groups	952.232	13	73.249		

Total	1012.500	14			

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCA
V3\DCA_Desktop_V3.sav'.
DATASET NAME DataSet7 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	534.405	1	534.405	1.594	.229
Within Groups	4358.929	13	335.302		
Total	4893.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	40.833	1	40.833	.109	.746
Within Groups	4852.500	13	373.269		
Total	4893.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1153.924	2	576.962	1.852	.199
Within Groups	3739.410	12	311.617		
Total	4893.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	575.030	1	575.030	1.731	.211
Within Groups	4318.304	13	332.177		
Total	4893.333	14			

SUS Web J overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		69.167
Median		75.000
Mode		100.0
Std. Deviation		27.1515
Minimum		7.5
Maximum		100.0
Sum		1037.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	7.5	1	6.7	6.7	6.7
	35.0	1	6.7	6.7	13.3
	40.0	1	6.7	6.7	20.0

	52.5	1	6.7	6.7	26.7
	55.0	1	6.7	6.7	33.3
	60.0	1	6.7	6.7	40.0
	72.5	1	6.7	6.7	46.7
	75.0	1	6.7	6.7	53.3
	82.5	2	13.3	13.3	66.7
	85.0	1	6.7	6.7	73.3
	90.0	1	6.7	6.7	80.0
	100.0	3	20.0	20.0	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCB
V3\DCB_Desktop_V3.sav'.
DATASET NAME DataSet34 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop J overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		67.167
Median		72.500
Mode		37.5
Std. Deviation		20.7422
Minimum		37.5
Maximum		100.0
Sum		1007.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	2	13.3	13.3	13.3
	40.0	1	6.7	6.7	20.0
	45.0	1	6.7	6.7	26.7

52.5	1	6.7	6.7	33.3
60.0	1	6.7	6.7	40.0
67.5	1	6.7	6.7	46.7
72.5	1	6.7	6.7	53.3
75.0	1	6.7	6.7	60.0
77.5	1	6.7	6.7	66.7
80.0	1	6.7	6.7	73.3
82.5	1	6.7	6.7	80.0
85.0	1	6.7	6.7	86.7
95.0	1	6.7	6.7	93.3
100.0	1	6.7	6.7	100.0
Total	15	100.0	100.0	

```
DATASET ACTIVATE DataSet33.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
DATASET CLOSE DataSet34.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web J and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0
Mean		68.333
Median		77.500
Mode		100.0
Std. Deviation		30.1762
Minimum		7.5
Maximum		100.0
Sum		820.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	7.5	1	8.3	8.3	8.3
	35.0	1	8.3	8.3	16.7
	40.0	1	8.3	8.3	25.0
	52.5	1	8.3	8.3	33.3
	55.0	1	8.3	8.3	41.7
	72.5	1	8.3	8.3	50.0
	82.5	1	8.3	8.3	58.3
	85.0	1	8.3	8.3	66.7
	90.0	1	8.3	8.3	75.0
	100.0	3	25.0	25.0	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		72.500
Median		75.000
Mode		60.0[b]
Std. Deviation		11.4564
Minimum		60.0
Maximum		82.5
Sum		217.5

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	60.0	1	33.3	33.3	33.3
	75.0	1	33.3	33.3	66.7
	82.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web J and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		76.875
Median		78.750
Mode		100.0
Std. Deviation		19.5827
Minimum		52.5
Maximum		100.0
Sum		615.0

a. Age = 18-24

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	12.5	12.5	12.5
	55.0	1	12.5	12.5	25.0
	60.0	1	12.5	12.5	37.5

	72.5	1	12.5	12.5	50.0
	85.0	1	12.5	12.5	62.5
	90.0	1	12.5	12.5	75.0
	100.0	2	25.0	25.0	100.0
	Total	8	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		73.125
Median		78.750
Mode		35.0[b]
Std. Deviation		27.4905
Minimum		35.0
Maximum		100.0
Sum		292.5

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	25.0	25.0	25.0
	75.0	1	25.0	25.0	50.0
	82.5	1	25.0	25.0	75.0
	100.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		61.250
Median		61.250

Mode	40.0[b]
Std. Deviation	30.0520
Minimum	40.0
Maximum	82.5
Sum	122.5

a. Age = 35-44

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	50.0	50.0	50.0
	82.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Age = 35-44

Age = 55-64

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		7.500
Median		7.500
Mode		7.5
Minimum		7.5
Maximum		7.5
Sum		7.5

a. Age = 55-64

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	7.5	1	100.0	100.0	100.0

a. Age = 55-64

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web J and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		79.375
Median		78.750
Mode		60.0[b]
Std. Deviation		16.6302
Minimum		60.0
Maximum		100.0
Sum		317.5

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	60.0	1	25.0	25.0	25.0
	75.0	1	25.0	25.0	50.0
	82.5	1	25.0	25.0	75.0
	100.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		50.833

Median	55.000
Mode	7.5[b]
Std. Deviation	41.4075
Minimum	7.5
Maximum	90.0
Sum	152.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	7.5	1	33.3	33.3	33.3
	55.0	1	33.3	33.3	66.7
	90.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	8
	Missing	0
Mean		70.938
Median		77.500
Mode		100.0
Std. Deviation		25.6674
Minimum		35.0
Maximum		100.0
Sum		567.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

Valid	35.0	1	12.5	12.5	12.5
	40.0	1	12.5	12.5	25.0
	52.5	1	12.5	12.5	37.5
	72.5	1	12.5	12.5	50.0
	82.5	1	12.5	12.5	62.5
	85.0	1	12.5	12.5	75.0
	100.0	2	25.0	25.0	100.0
	Total	8	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web J and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		72.500
Median		85.000
Mode		100.0
Std. Deviation		33.1662
Minimum		7.5
Maximum		100.0
Sum		507.5

a. Occupation = 1

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	7.5	1	14.3	14.3	14.3

	52.5	1	14.3	14.3	28.6
	72.5	1	14.3	14.3	42.9
	85.0	1	14.3	14.3	57.1
	90.0	1	14.3	14.3	71.4
	100.0	2	28.6	28.6	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		64.583
Median		65.000
Mode		35.0[b]
Std. Deviation		25.5155
Minimum		35.0
Maximum		100.0
Sum		387.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	35.0	1	16.7	16.7	16.7
	40.0	1	16.7	16.7	33.3
	55.0	1	16.7	16.7	50.0
	75.0	1	16.7	16.7	66.7
	82.5	1	16.7	16.7	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other area)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		82.500
Median		82.500
Mode		82.5
Minimum		82.5
Maximum		82.5
Sum		82.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	82.5	1	100.0	100.0	100.0

a. Occupation = 4

Occupation = Other Occupation

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		60.000
Median		60.000
Mode		60.0
Minimum		60.0
Maximum		60.0
Sum		60.0

a. Occupation = 5

SUS_Total[a]

	Frequency	Percent	Valid Percent	Cumulative Percent

Valid	60.0	1	100.0	100.0	100.0

a. Occupation = 5

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCB
V3\DCB_Desktop_V3.sav'.
DATASET NAME DataSet35 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS desktop J and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0
Mean		68.542
Median		76.250
Mode		37.5
Std. Deviation		22.7750
Minimum		37.5
Maximum		100.0
Sum		822.5

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	2	16.7	16.7	16.7
	40.0	1	8.3	8.3	25.0
	45.0	1	8.3	8.3	33.3
	67.5	1	8.3	8.3	41.7
	75.0	1	8.3	8.3	50.0
	77.5	1	8.3	8.3	58.3

	80.0	1	8.3	8.3	66.7
	82.5	1	8.3	8.3	75.0
	85.0	1	8.3	8.3	83.3
	95.0	1	8.3	8.3	91.7
	100.0	1	8.3	8.3	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		61.667
Median		60.000
Mode		52.5[b]
Std. Deviation		10.1036
Minimum		52.5
Maximum		72.5
Sum		185.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	33.3	33.3	33.3
	60.0	1	33.3	33.3	66.7
	72.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS desktop J and Age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		65.833
Median		75.000
Mode		37.5[b]
Std. Deviation		21.9730
Minimum		37.5
Maximum		95.0
Sum		592.5

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	11.1	11.1	11.1
	40.0	1	11.1	11.1	22.2
	45.0	1	11.1	11.1	33.3
	52.5	1	11.1	11.1	44.4
	75.0	1	11.1	11.1	55.6
	80.0	1	11.1	11.1	66.7
	82.5	1	11.1	11.1	77.8
	85.0	1	11.1	11.1	88.9
	95.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		66.667
Median		67.500
Mode		60.0[b]
Std. Deviation		6.2915
Minimum		60.0
Maximum		72.5
Sum		200.0

a. Age = 25-34

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	60.0	1	33.3	33.3	33.3
	67.5	1	33.3	33.3	66.7
	72.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		71.667
Median		77.500
Mode		37.5[b]
Std. Deviation		31.6557
Minimum		37.5

Maximum	100.0
Sum	215.0

a. Age = 35-44

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	33.3	33.3	33.3
	77.5	1	33.3	33.3	66.7
	100.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Age = 35-44

```
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS desktop J and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		61.667
Median		60.000
Mode		52.5[b]
Std. Deviation		10.1036
Minimum		52.5
Maximum		72.5
Sum		185.0

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	33.3	33.3	33.3
	60.0	1	33.3	33.3	66.7
	72.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		55.833
Median		45.000
Mode		40.0[b]
Std. Deviation		23.2289
Minimum		40.0
Maximum		82.5
Sum		167.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	33.3	33.3	33.3
	45.0	1	33.3	33.3	66.7
	82.5	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	9
	Missing	0
Mean		72.778
Median		77.500
Mode		37.5
Std. Deviation		22.3063
Minimum		37.5
Maximum		100.0
Sum		655.0

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	2	22.2	22.2	22.2
	67.5	1	11.1	11.1	33.3
	75.0	1	11.1	11.1	44.4
	77.5	1	11.1	11.1	55.6
	80.0	1	11.1	11.1	66.7
	85.0	1	11.1	11.1	77.8
	95.0	1	11.1	11.1	88.9
	100.0	1	11.1	11.1	100.0
	Total	9	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS desktop J and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		71.429
Median		80.000
Mode		37.5[b]
Std. Deviation		21.5956
Minimum		37.5
Maximum		95.0
Sum		500.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	14.3	14.3	14.3
	45.0	1	14.3	14.3	28.6
	75.0	1	14.3	14.3	42.9
	80.0	1	14.3	14.3	57.1
	82.5	1	14.3	14.3	71.4
	85.0	1	14.3	14.3	85.7
	95.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		63.750
Median		63.750
Mode		37.5[b]
Std. Deviation		23.5982
Minimum		37.5
Maximum		100.0
Sum		382.5

a. Occupation = 2

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	16.7	16.7	16.7
	40.0	1	16.7	16.7	33.3
	60.0	1	16.7	16.7	50.0
	67.5	1	16.7	16.7	66.7
	77.5	1	16.7	16.7	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time (other are)

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		72.500
Median		72.500
Mode		72.5
Minimum		72.5

Maximum	72.5
Sum	72.5

a. Occupation = 4

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	1	100.0	100.0	100.0

a. Occupation = 4

Occupation = Other Occupation

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		52.500
Median		52.500
Mode		52.5
Minimum		52.5
Maximum		52.5
Sum		52.5

a. Occupation = 5

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	52.5	1	100.0	100.0	100.0

a. Occupation = 5

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCB
V3\DCB_Web_V3.sav'.
DATASET NAME DataSet26 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for A Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	41.667	1	41.667	.053	.822
Within Groups	10279.167	13	790.705		
Total	10320.833	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4466.146	3	1488.715	2.797	.090
Within Groups	5854.688	11	532.244		
Total	10320.833	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1450.260	2	725.130	.981	.403
Within Groups	8870.573	12	739.214		
Total	10320.833	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	465.625	3	155.208	.173	.912
Within Groups	9855.208	11	895.928		
Total	10320.833	14			

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet26.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRDCB
V3\DCB_Desktop_V3.sav'.
DATASET NAME DataSet27 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	113.438	1	113.438	.250	.626
Within Groups	5909.896	13	454.607		
Total	6023.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	77.500	2	38.750	.078	.925
Within Groups	5945.833	12	495.486		
Total	6023.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	759.444	2	379.722	.866	.445
Within Groups	5263.889	12	438.657		
Total	6023.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	440.744	3	146.915	.289	.832
Within Groups	5582.589	11	507.508		
Total	6023.333	14			

SUS Web B overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		85.167
Median		92.500
Mode		97.5
Minimum		40.0

Maximum	100.0
Sum	1277.5

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	6.7	6.7	6.7
	72.5	2	13.3	13.3	20.0
	75.0	1	6.7	6.7	26.7
	77.5	1	6.7	6.7	33.3
	82.5	2	13.3	13.3	46.7
	92.5	2	13.3	13.3	60.0
	97.5	4	26.7	26.7	86.7
	100.0	2	13.3	13.3	100.0
	Total	15	100.0	100.0	

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Desktop_V3.sav'.
DATASET NAME DataSet23 WINDOW=FRONT.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop B overall

Statistics

SUS_Total

N	Valid	15
	Missing	0
Mean		70.667
Median		70.000
Mode		70.0[a]
Minimum		25.0
Maximum		100.0
Sum		1060.0

a. Multiple modes exist. The smallest value is shown

SUS_Total

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	6.7	6.7	6.7
	37.5	1	6.7	6.7	13.3
	42.5	1	6.7	6.7	20.0
	45.0	1	6.7	6.7	26.7
	62.5	1	6.7	6.7	33.3
	67.5	1	6.7	6.7	40.0
	70.0	2	13.3	13.3	53.3
	77.5	1	6.7	6.7	60.0
	87.5	1	6.7	6.7	66.7
	90.0	2	13.3	13.3	80.0
	97.5	2	13.3	13.3	93.3
	100.0	1	6.7	6.7	100.0
	Total	15	100.0	100.0	

SUS web B and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		82.143
Median		82.500
Mode		72.5[b]
Minimum		72.5
Maximum		97.5
Sum		575.0

a. Age = 18-24

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	2	28.6	28.6	28.6
	75.0	1	14.3	14.3	42.9
	82.5	2	28.6	28.6	71.4
	92.5	1	14.3	14.3	85.7
	97.5	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		93.000
Median		97.500
Mode		97.5
Minimum		77.5
Maximum		100.0
Sum		465.0

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	20.0	20.0	20.0
	92.5	1	20.0	20.0	40.0
	97.5	2	40.0	40.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		100.000
Median		100.000
Mode		100.0
Minimum		100.0
Maximum		100.0
Sum		100.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	100.0	1	100.0	100.0	100.0

a. Age = 35-44

Age = 55-64

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		68.750
Median		68.750
Mode		40.0[b]
Minimum		40.0
Maximum		97.5
Sum		137.5

a. Age = 55-64

b. Multiple modes exist. The
smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	50.0	50.0	50.0
	97.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Age = 55-64

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Desktop_V3.sav'.
DATASET NAME DataSet41 WINDOW=FRONT.
SORT CASES  BY Age.
SPLIT FILE SEPARATE BY Age.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop B and age

Age = 18-24

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		67.083
Median		68.750
Mode		42.5[b]
Minimum		42.5
Maximum		90.0
Sum		402.5

a. Age = 18-24

b. Multiple modes exist. The
smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	16.7	16.7	16.7
	45.0	1	16.7	16.7	33.3
	67.5	1	16.7	16.7	50.0
	70.0	1	16.7	16.7	66.7
	87.5	1	16.7	16.7	83.3
	90.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 18-24

Age = 25-34

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		87.500
Median		93.750
Mode		97.5
Minimum		62.5
Maximum		100.0
Sum		525.0

a. Age = 25-34

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	62.5	1	16.7	16.7	16.7
	77.5	1	16.7	16.7	33.3
	90.0	1	16.7	16.7	50.0
	97.5	2	33.3	33.3	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Age = 25-34

Age = 35-44

Statistics[a]

SUS_Total

N	Valid	1
	Missing	0
Mean		70.000
Median		70.000
Mode		70.0
Minimum		70.0
Maximum		70.0
Sum		70.0

a. Age = 35-44

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	70.0	1	100.0	100.0	100.0

a. Age = 35-44

Age = 55-64

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		31.250
Median		31.250
Mode		25.0[b]
Minimum		25.0
Maximum		37.5
Sum		62.5

a. Age = 55-64

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	50.0	50.0	50.0
	37.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Age = 55-64

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Web_V3.sav'.
DATASET NAME DataSet1 WINDOW=FRONT.
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS web B and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		88.125
Median		90.000
Mode		72.5[b]
Minimum		72.5
Maximum		100.0
Sum		352.5

a. Pro_Exp = No Experience

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	1	25.0	25.0	25.0
	82.5	1	25.0	25.0	50.0
	97.5	1	25.0	25.0	75.0
	100.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		87.500
Median		92.500
Mode		72.5[b]
Minimum		72.5
Maximum		100.0
Sum		437.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	1	20.0	20.0	20.0
	75.0	1	20.0	20.0	40.0
	92.5	1	20.0	20.0	60.0
	97.5	1	20.0	20.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		81.250
Median		87.500
Mode		97.5
Minimum		40.0
Maximum		97.5
Sum		487.5

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	16.7	16.7	16.7
	77.5	1	16.7	16.7	33.3
	82.5	1	16.7	16.7	50.0
	92.5	1	16.7	16.7	66.7
	97.5	2	33.3	33.3	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = More than 1 year

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Desktop_V3.sav'.
DATASET NAME DataSet2 WINDOW=FRONT.
SORT CASES  BY Pro_Exp.
SPLIT FILE SEPARATE BY Pro_Exp.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop B and Experience

No Experience

Statistics[a]

SUS_Total

N	Valid	4
	Missing	0
Mean		68.125
Median		70.000
Mode		70.0
Minimum		42.5
Maximum		90.0
Sum		272.5

a. Pro_Exp = No Experience

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	25.0	25.0	25.0
	70.0	2	50.0	50.0	75.0
	90.0	1	25.0	25.0	100.0
	Total	4	100.0	100.0	

a. Pro_Exp = No Experience

Year or less than 1 year

Statistics[a]

SUS_Total

N	Valid	5
	Missing	0
Mean		62.500
Median		62.500
Mode		37.5[b]
Minimum		37.5
Maximum		100.0
Sum		312.5

a. Pro_Exp = Year or less than 1 year

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	20.0	20.0	20.0
	45.0	1	20.0	20.0	40.0
	62.5	1	20.0	20.0	60.0
	67.5	1	20.0	20.0	80.0
	100.0	1	20.0	20.0	100.0
	Total	5	100.0	100.0	

a. Pro_Exp = Year or less than 1 year

More than 1 year

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		79.167
Median		88.750
Mode		97.5
Minimum		25.0
Maximum		97.5
Sum		475.0

a. Pro_Exp = More than 1 year

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	16.7	16.7	16.7
	77.5	1	16.7	16.7	33.3
	87.5	1	16.7	16.7	50.0
	90.0	1	16.7	16.7	66.7
	97.5	2	33.3	33.3	100.0
	Total	6	100.0	100.0	

a. Pro_Exp = More than 1 year

```
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web B and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	13
	Missing	0
Mean		84.231
Median		92.500
Mode		97.5
Std. Deviation		16.9676
Minimum		40.0
Maximum		100.0
Sum		1095.0

a. Gender = Male

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	7.7	7.7	7.7

	72.5	2	15.4	15.4	23.1
	75.0	1	7.7	7.7	30.8
	77.5	1	7.7	7.7	38.5
	82.5	1	7.7	7.7	46.2
	92.5	2	15.4	15.4	61.5
	97.5	4	30.8	30.8	92.3
	100.0	1	7.7	7.7	100.0
	Total	13	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		91.250
Median		91.250
Mode		82.5[b]
Std. Deviation		12.3744
Minimum		82.5
Maximum		100.0
Sum		182.5

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	82.5	1	50.0	50.0	50.0
	100.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Web B and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		84.286
Median		82.500
Mode		72.5[b]
Std. Deviation		11.4304
Minimum		72.5
Maximum		97.5
Sum		590.0

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	72.5	2	28.6	28.6	28.6
	75.0	1	14.3	14.3	42.9
	82.5	1	14.3	14.3	57.1
	92.5	1	14.3	14.3	71.4
	97.5	2	28.6	28.6	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		91.250
Median		95.000
Mode		97.5
Std. Deviation		9.1856
Minimum		77.5
Maximum		100.0
Sum		547.5

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	77.5	1	16.7	16.7	16.7
	82.5	1	16.7	16.7	33.3
	92.5	1	16.7	16.7	50.0
	97.5	2	33.3	33.3	83.3
	100.0	1	16.7	16.7	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = Full time(other area)

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		70.000
Median		70.000
Mode		40.0[b]
Std. Deviation		42.4264
Minimum		40.0
Maximum		100.0
Sum		140.0

a. Occupation = 4

b. Multiple modes exist. The

smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	40.0	1	50.0	50.0	50.0
	100.0	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Occupation = 4

```
DATASET ACTIVATE DataSet5.
DATASET CLOSE DataSet9.
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Desktop_V3.sav'.
DATASET NAME DataSet10 WINDOW=FRONT.
SORT CASES  BY Gender.
SPLIT FILE SEPARATE BY Gender.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop B and Gender

Gender = Male

Statistics[a]

SUS_Total

N	Valid	12
	Missing	0
Mean		71.250
Median		73.750
Mode		70.0[b]
Std. Deviation		24.0619
Minimum		25.0
Maximum		97.5

Sum	855.0

a. Gender = Male

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	8.3	8.3	8.3
	37.5	1	8.3	8.3	16.7
	45.0	1	8.3	8.3	25.0
	67.5	1	8.3	8.3	33.3
	70.0	2	16.7	16.7	50.0
	77.5	1	8.3	8.3	58.3
	87.5	1	8.3	8.3	66.7
	90.0	2	16.7	16.7	83.3
	97.5	2	16.7	16.7	100.0
	Total	12	100.0	100.0	

a. Gender = Male

Gender = Female

Statistics[a]

SUS_Total

N	Valid	3
	Missing	0
Mean		68.333
Median		62.500
Mode		42.5[b]
Std. Deviation		29.1905
Minimum		42.5
Maximum		100.0
Sum		205.0

a. Gender = Female

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	33.3	33.3	33.3
	62.5	1	33.3	33.3	66.7
	100.0	1	33.3	33.3	100.0
	Total	3	100.0	100.0	

a. Gender = Female

```
SORT CASES  BY Occupation.
SPLIT FILE SEPARATE BY Occupation.
FREQUENCIES VARIABLES=SUS_Total
  /STATISTICS=STDDEV MINIMUM MAXIMUM MEAN MEDIAN MODE SUM
  /ORDER=ANALYSIS.
```

SUS Desktop B and Occupation

Occupation = Undergraduate

Statistics[a]

SUS_Total

N	Valid	7
	Missing	0
Mean		71.071
Median		70.000
Mode		37.5[b]
Std. Deviation		23.4013
Minimum		37.5
Maximum		100.0
Sum		497.5

a. Occupation = 1

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	37.5	1	14.3	14.3	14.3
	45.0	1	14.3	14.3	28.6
	67.5	1	14.3	14.3	42.9
	70.0	1	14.3	14.3	57.1
	87.5	1	14.3	14.3	71.4
	90.0	1	14.3	14.3	85.7
	100.0	1	14.3	14.3	100.0
	Total	7	100.0	100.0	

a. Occupation = 1

Occupation = Postgraduate

Statistics[a]

SUS_Total

N	Valid	6
	Missing	0
Mean		79.167
Median		83.750
Mode		97.5
Std. Deviation		21.0753
Minimum		42.5
Maximum		97.5
Sum		475.0

a. Occupation = 2

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	42.5	1	16.7	16.7	16.7
	70.0	1	16.7	16.7	33.3
	77.5	1	16.7	16.7	50.0
	90.0	1	16.7	16.7	66.7

	97.5	2	33.3	33.3	100.0
	Total	6	100.0	100.0	

a. Occupation = 2

Occupation = full time (other area)

Statistics[a]

SUS_Total

N	Valid	2
	Missing	0
Mean		43.750
Median		43.750
Mode		25.0[b]
Std. Deviation		26.5165
Minimum		25.0
Maximum		62.5
Sum		87.5

a. Occupation = 4

b. Multiple modes exist. The smallest value is shown

SUS_Total[a]

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	25.0	1	50.0	50.0	50.0
	62.5	1	50.0	50.0	100.0
	Total	2	100.0	100.0	

a. Occupation = 4

```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3\PRCSA
V3\A_Web_V3.sav'.
DATASET NAME DataSet11 WINDOW=FRONT.
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

ANOVA Testing for B Group Applications

Web Application

Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	85.401	1	85.401	.308	.589
Within Groups	3607.933	13	277.533		
Total	3693.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1129.851	3	376.617	1.616	.242
Within Groups	2563.482	11	233.044		
Total	3693.333	14			

```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	154.271	2	77.135	.262	.774
Within Groups	3539.063	12	294.922		
Total	3693.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	687.530	2	343.765	1.372	.291
Within Groups	3005.804	12	250.484		
Total	3693.333	14			

```
ONEWAY SUS_Total BY Gender
  /MISSING ANALYSIS.
```

Desktop Application
Gender

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	20.417	1	20.417	.033	.859
Within Groups	8072.917	13	620.994		
Total	8093.333	14			

```
ONEWAY SUS_Total BY Age
  /MISSING ANALYSIS.
```

Age

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	4885.000	3	1628.333	5.583	.014
Within Groups	3208.333	11	291.667		

Total	8093.333	14			

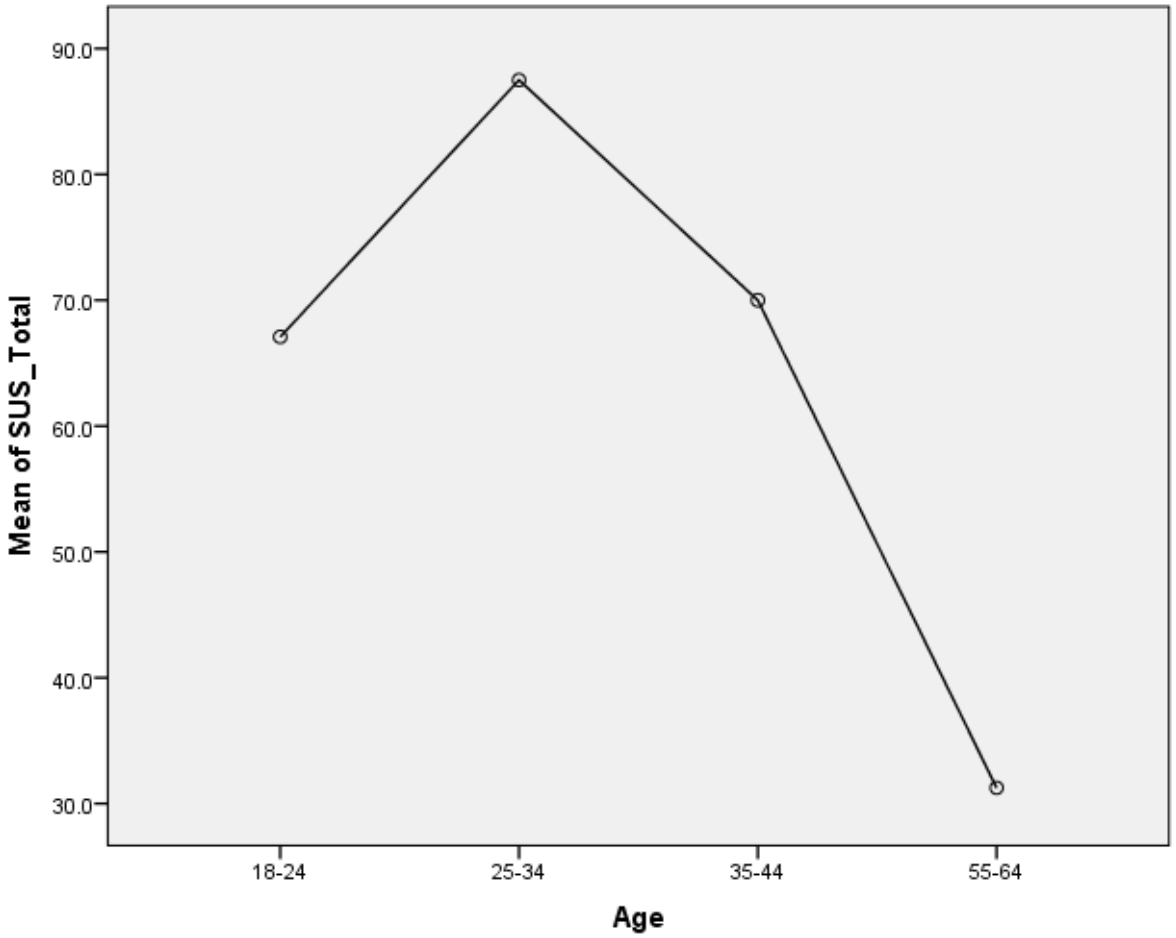
```
GET
  FILE='\\RAIDVIZ\share\fAlmansour\_Fahad\dEv Experminte\Second
Experiment\SUS DATA FOR THE FINAL ANALYSIS\Data ready for analysis\V3 used
for the study results\PRCSA V3\A_Desktop_V3.sav'.
DATASET NAME DataSet1 WINDOW=FRONT.
ONEWAY SUS_Total BY Age
  /STATISTICS DESCRIPTIVES
  /PLOT MEANS
  /MISSING ANALYSIS
  /POSTHOC=LSD ALPHA(0.05).
```

Descriptives

SUS_Total

	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
18-24	6	67.083	20.2124	8.2517	45.872	88.295	42.5	90.0
25-34	6	87.500	14.7479	6.0208	72.023	102.977	62.5	100.0
35-44	1	70.000	.	.	.	.	70.0	70.0
55-64	2	31.250	8.8388	6.2500	-48.164	110.664	25.0	37.5
Total	15	70.667	24.0436	6.2080	57.352	83.982	25.0	100.0

Means Plots



Oneway

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	3883.958	2	1941.979	5.536	.020
Within Groups	4209.375	12	350.781		
Total	8093.333	14			

Post Hoc Tests

Multiple Comparisons

Dependent Variable: SUS_Total

Tukey HSD

(I) For_ANOVA_Test_Only	(J) For_ANOVA_Test_Only	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
18-24	25-34	-20.4167	10.8133	.184	-49.265	8.432
	35-44	22.9167	13.2435	.234	-12.415	58.249
25-34	18-24	20.4167	10.8133	.184	-8.432	49.265
	35-44	43.3333*	13.2435	.017	8.001	78.665
35-44	18-24	-22.9167	13.2435	.234	-58.249	12.415
	25-34	-43.3333*	13.2435	.017	-78.665	-8.001

*. The mean difference is significant at the 0.05 level.

Homogeneous Subsets

SUS_Total

Tukey HSD[a,b]

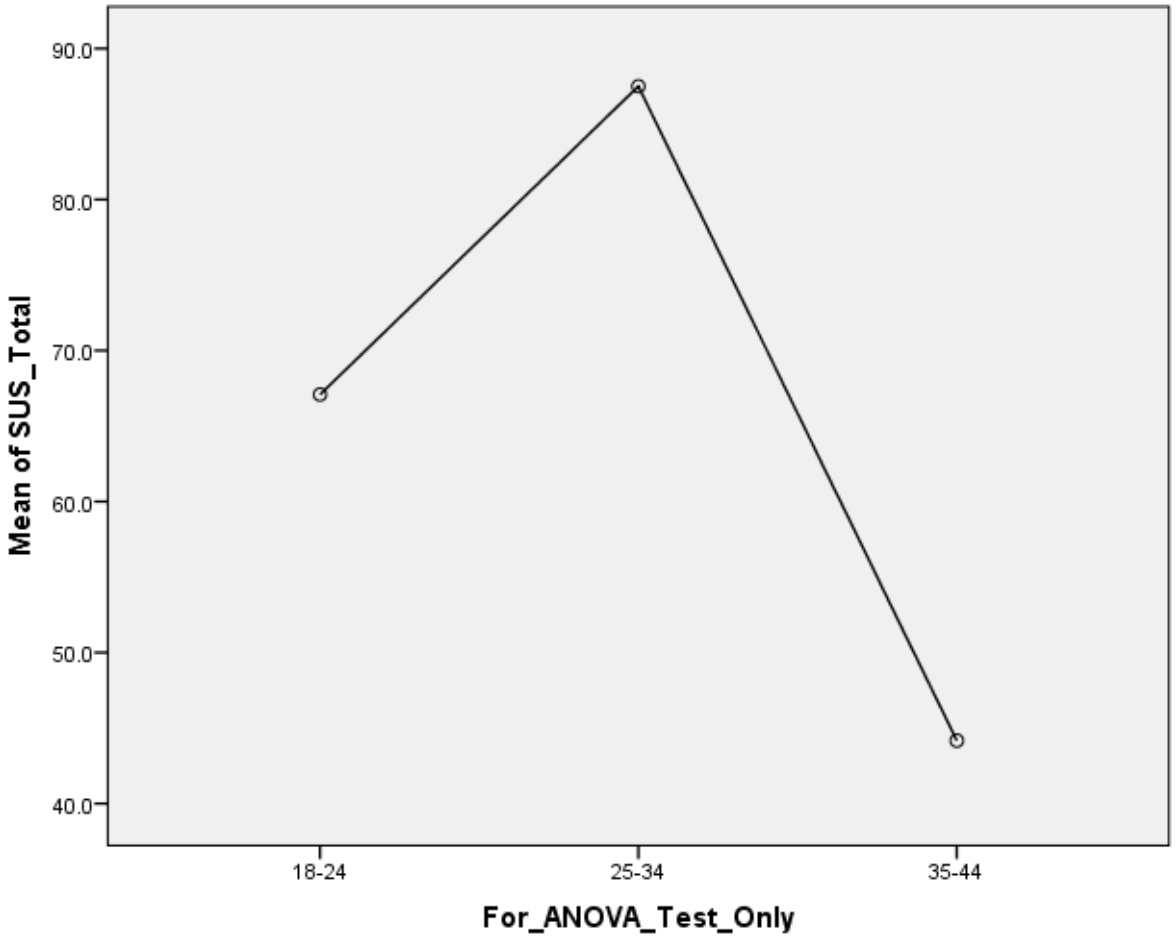
For_ANOVA_Test_Only	N	Subset for alpha = 0.05	
		1	2
35-44	3	44.167	
18-24	6	67.083	67.083
25-34	6		87.500
Sig.		.200	.269

Means for groups in homogeneous subsets are displayed.

a. Uses Harmonic Mean Sample Size = 4.500.

b. The group sizes are unequal. The harmonic mean of the group sizes is used. Type I error levels are not guaranteed.

Means Plots




```
ONEWAY SUS_Total BY Pro_Exp
  /MISSING ANALYSIS.
```

Experience

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	792.813	2	396.406	.652	.539
Within Groups	7300.521	12	608.377		
Total	8093.333	14			

```
ONEWAY SUS_Total BY Occupation
  /MISSING ANALYSIS.
```

Occupation

ANOVA

SUS_Total

	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	1883.661	2	941.830	1.820	.204
Within Groups	6209.673	12	517.473		
Total	8093.333	14			

References

Aberg, J., & Shahmehri, N. (2000). The Role of Human Web Assistants in E-Commerce: An Analysis and a Usability Study. *Internet Research*, *10*(2), 114–125.

Abras, C., Maloney-Krichmar, D., & Preece, J. (2004). User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications*, *37*(4), 445–456.

Administrator, A. E. A. (2014). American Evaluation Association. Retrieved July 14, 2015, from http://www.eval.org/p/bl/et/blogid=2

Aflafla1. (2014). Iterative development model. Retrieved October 18, 2015, from https://commons.wikimedia.org/wiki/File:Iterative_development_model.svg

Alfrih, F. M. (2010). *The role of academic libraries in supporting distance learning in Saudi higher education: a case study approach*. Loughborough University. Retrieved from https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/6935

Alhalalat, S. I. (2005). *Information flow in virtual organisations*. Loughborough University.

Alkhuraiji, A., Liu, S., Oderanti, F. O., Annansingh, F., & Pan, J. (2014). Knowledge network modelling to support decision-making for strategic intervention in IT project-oriented change management. *Journal of Decision Systems*, *23*(3), 285–302.

Alkhuraiji, A., Liu, S., Oderanti, F. O., & Megicks, P. (2015). New structured knowledge network for strategic decision-making in IT innovative and implementable projects. *Journal of Business Research*.

Alkraiji, A. (2012). *Issues of the Adoption of HIT related Standards at the Decision-Making Stage of Six Tertiary Healthcare Organisations in Saudi Arabia*. Loughborough University. Retrieved from https://dspace.lboro.ac.uk/2134/9848

Almansour, F., & Stuart, L. (2014). Promoting the use of Design Evaluation Techniques within Software Development. In *BCS HCI*.

Alwan, M. (2015). What is System Development Life Cycle? Retrieved October 29, 2015, from https://airbrake.io/blog/insight/what-is-system-development-life-cycle

Anderson, T. (2004). Teaching in an online learning context. In *Theory and practice of online learning* (2nd ed.). Athabasca University Press.

Appana, S. (2008). A review of benefits and limitations of online learning in the context of the student, the instructor, and the tenured faculty. *International Journal on ELearning*, *7*(1).

Ardito, C., Buono, P., & Caivano, D. (2014). Investigating and promoting UX practice in industry: An experimental study. *International Journal of Human-Computer Studies*, *72*(6), 542–551.

Ardito, C., Buono, P., Caivano, D., Costabile, M. F., Lanzilotti, R., Bruun, A., & Stage, J. (2011). Usability evaluation: A survey of software development organizations. In *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering (SEKE'2011), Eden Roc Renaissance, Miami Beach, USA, July 7-9, 2011* (pp. 282–287).

Ardito, C., Costabile, M. F., De Angeli, A., & Lanzilotti, R. (2006). Systematic evaluation of e-learning systems. In *Proceedings of the 4th Nordic conference on Human-computer interaction changing roles - NordiCHI '06* (pp. 195–202). New York, New York, USA: ACM Press.

Ardito, C., Costabile, M. F., Marsico, M. De, Lanzilotti, R., Levialdi, S., Roselli, T., & Rossano, V. (2006). An approach to usability evaluation of e-learning applications. *Universal Access in the Information Society*, *4*(3), 270–283.

Bak, J. O., Nguyen, K., Risgaard, P., & Stage, J. (2008). Obstacles to usability evaluation in practice. In *Proceedings of the 5th Nordic conference on Human-computer interaction building bridges - NordiCHI '08* (p. 23). New York, New York, USA: ACM Press.

Bandura, A. (1994). Self-efficacy. In V. S. Ramachaudran (Ed.). *Encyclopedia of Human Behavior, , 4*, 71–81.

Bangor, A., Kortum, P. T., & Miller, J. T. (2008). An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction*, *24*(6), 574–594.

Bargas-Avila, J. A., Lötscher, J., Orsini, S., & Opwis, K. (2009). Intranet satisfaction questionnaire: Development and validationof a questionnaire to measure user satisfaction with the Intranet. *Computers in Human Behavior*, *25*(6), 1241–1250.

Bassil, Y. (2012). A Simulation Model for the Waterfall Software Development Life Cycle. *Internation Journal of Engineering & Technology(iJET)*, *2*(5).

Beaudouin-Lafon, M. (2004). Designing interaction, not interfaces. In *Proceedings of the working conference on Advanced visual interfaces - AVI '04* (pp. 15–22). New York, New York, USA: ACM Press.

Beck, K., & Andres, C. (2004). *Extreme Programming Explained: Embrace Change* (2nd ed.). Addison-Wesley.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Principles behind the Agile Manifesto. Retrieved September 4,

2015, from http://www.agilemanifesto.org/principles.html

Benyon, D. (2010). *Designing interactive systems A comprehensive guide to HCI and interaction design* (2nd ed.). Addison Wesley.

Bevan, N. (1999). *Usability Issues in Web Site Design*. *HCI*. Retrieved from http://www.npl.co.uk/npl/sections/us

Bevan, N. (2005). Cost-Benefit Framework and Case Studies. In R. G. Bias & D. J. Mayhew (Eds.), *Cost-Justifying Usability An Update for an Internet Age* (2nd ed., pp. 575–600). Elsevier.

Bevan, N. (2006). Practical issues in usability measurement. *Interactions*, *13*(6), 42.

Bevan, N. (2008). Classifying and selecting UX and usability measures. In *International Workshop on Meaningful Measures: Valid Useful User Experience Measurement (VUUM)* (pp. 13–18).

Bias, R. G., & Mayhew, D. J. (2005). *Cost-justifying usability: an update for an Internet age* (2nd ed.). Elsevier.

Boiano, S., Bowen, J., & Gaia, G. (2012). Usability, design and content issues of mobile apps for cultural heritage promotion: The Malta culture guide experience. *arXiv Preprint arXiv:1207.3422*.

Bowler, L., Koshman, S., Oh, J. S., He, D., Callery, B., Bowker, G., & Cox, R. (2011). Issues in User-Centered Design in LIS. *Library Trends*, *59*(4), 721–725.

Bowman, D. A., Gabbard, J. L., & Hix, D. (2002). A Survey of Usability Evaluation in Virtual Environments: Classification and Comparison of Methods. *Presence: Teleoperators and Virtual Environments*, *11*(4), 404–424.

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101.

Brézillon, P., Borges, M. R. S., Pino, J. A., & Pomerol, J.-C. (2008). Lessons Learned from Three Case Studies. *Journal of Decision Systems*, *17*(1), 27–40. http://doi.org/10.3166/jds.17.27-40

Brhel, M., Meth, H., Maedche, A., & Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology*, *61*, 163–181.

Brinck, T., Gergle, D., & Wood, S. D. (2002). *Designing Web Sites that Work: Usability for the Web*. Morgan Kaufmann.

Brooke, J. (1996). SUS: A “quick and dirty” usability scale. In *Usability Evaluation In Industry* (pp. 189–194). Taylor & Francis.

Brooke, J. (2013). SUS: a retrospective. *Journal of Usability Studies*, *8*(2), 29–40.

Bruun, A., & Stage, J. (2014). Barefoot usability evaluations. *Behaviour & Information Technology*, *33*(11), 1148–1167.

Bryman, A. (2006). Integrating quantitative and qualitative research: how is it done? *Qualitative Research*, *6*(1), 97–113.

Bryman, A. (2007). Barriers to Integrating Quantitative and Qualitative Research. *Journal of Mixed Methods Research*, *1*(1), 8–22.

Bryman, A. (2008). *Social research methods*. Oxford University Press.

Burrell, A., & Sodan, A. C. (2006). Web Interface Navigation Design: Which Style of Navigation-Link Menus Do Users Prefer? In *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on Data Engineering Workshops* (pp. 1–10).

Calikli, G., Aslan, B., & Bener, A. (2010). Confirmation bias in software development and testing: An analysis of the effects of company size, experience and reasoning skills. In *Workshop on Psychology of Programming Interest Group (PPIG)* (p. 14). Leganes, Madrid, Spain.

Campbell, A. (2013). Multum Non Multa. Retrieved January 25, 2017, from https://www.memoriapress.com/articles/multum-non-multa/

Carliner, S. (2004). *An overview of online learning*. Human Resource Development.

Carroll, J. M. (2014). Human Computer Interaction - brief intro. In *The Encyclopedia of Human-Computer Interaction* (2nd ed.). The Interaction Design Foundation. Retrieved from https://www.interaction-design.org/encyclopedia/human_computer_interaction_hci.html

Cburnett. (2005). Sample web form. Retrieved April 12, 2016, from https://commons.wikimedia.org/wiki/File:Sample_web_form.png

Ccit333. (2015). The Ten Minute Rule. Retrieved September 21, 2015, from https://ccit333.wikispaces.com/The+Ten+Minute+Rule

Chamberlain, S., Sharp, H., & Maiden, N. (2006). Towards a framework for integrating agile development and user-centred design. In *Extreme Programming and Agile Processes in Software Engineering* (pp. 143–153). Springer Berlin Heidelberg.

Chisholm, W., Vanderheiden, G., & Jacobs, I. (2001). Web content accessibility guidelines 1.0. *Interactions*, *8*(4), 35–54.

Churchill, E. F., Bowser, A., & Preece, J. (2013). Teaching and learning human-computer interaction. *Interactions*, *20*(2), 44.

coggnac.com. (2016). Bobby Approved: Practical Web Accessibility Evaluation Tool Icon. Retrieved October 30, 2016, from http://www.coggan.com/bobby-approved.html

Cohn, M. (2005). Scrum Overview for Agile Software Development. Retrieved October 25, 2015, from https://www.mountaingoatsoftware.com/agile/scrum/overview

Cohn, M. (2015a). Scrum. Retrieved October 25, 2015, from https://www.mountaingoatsoftware.com/agile/scrum

Cohn, M. (2015b). Sprint Review Meeting. Retrieved October 25, 2015, from https://www.mountaingoatsoftware.com/agile/scrum/sprint-review-meeting

Cohn, M. (2015c). User Stories. Retrieved October 25, 2015, from http://www.mountaingoatsoftware.com/agile/user-stories

Collis, J., & Hussey, R. (2013). *Business research: A practical guide for undergraduate and postgraduate students*. Palgrave macmillan.

Conrad, D. (2002). Deep in the hearts of learners: Insights into the nature of online community. *International Journal of E-Learning & Distance*, *17*(1), 1–19.

Cook, D. A., & Dupras, D. M. (2004). A practical guide to developing effective web-based learning. *Journal of General Internal Medicine*, *19*(6), 698–707.

Cornford, T., & Smithson, S. (2006). *Project research in information systems: a student's guide*. Palgrave.

Courage, C., & Baxter, K. (2005). *Understanding your users : a practical guide to user requirements methods, tools and techniques*. Morgan Kaufmann Publishers.

Creswell, J. W. (2003). *Research Design : Qualitative, Quantitative and Mixed Methods Approaches*. SAGE Publications.

Creswell, J. W. (2013). *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications.

Debnath, I., Hussain, M. M., Islam, T., Hossain, R., & Rahman, S. (2012). *The Future of Human Computer Interaction*. *School of Business, North South University*.

Dicks, R. S. (2002). Mis-usability: on the uses and misuses of usability testing. In *Proceedings of the 20th annual international conference on Computer documentation - SIGDOC '02* (pp. 26–30). New York, New York, USA: ACM Press.

dictionary.cambridge.org. (2017). pedagogy Meaning in the Cambridge English Dictionary. Retrieved January 25, 2017, from http://dictionary.cambridge.org/dictionary/english/pedagogy

Dix, A. (2011). Are five users enough? Retrieved December 22, 2015, from http://alandix.com/blog/2011/06/04/are-five-users-enough/

Dix, A., Finlay, J., Abwod, G. D., & Beale, R. (2004). *Human Computer Interaction* (3rd ed.). USA: Prentice Hall.

Dooley, R. (2013). How Warren Buffett Avoids Getting Trapped by Confirmation Bias. Retrieved from http://www.forbes.com/sites/rogerdooley/2013/05/07/buffett-confirmation-bias/#66ebb3a23a83

Dumas, J. S., & Redish, J. (1999). *A Practical Guide to Usability Testing* (2nd Revise). University of Chicago Press.

Easterby-Smith, M., Thorpe, R., & Lowe, A. (2002). *Management research : an introduction* (2nd ed.). SAGE.

Eklund, J., & Levingston, C. (2008). Usability in Agile development. In *UX Research* (pp. 1–7).

Eldredge, J. D. (2004). Inventory of research methods for librarianship and informatics. *Journal of the Medical Library Association : JMLA*, *92*(1), 83–90.

Extremeprogramming. (2013). Extreme Programming: A Gentle Introduction. Retrieved September 8, 2015, from http://www.extremeprogramming.org/

Extremeprogramming.org. (2009). Extreme Programming Values. Retrieved September 25, 2015, from http://www.extremeprogramming.org/values.html

Farrell, S. (2015). Utility Navigation: What It Is and How to Design It. Retrieved November 10, 2015, from http://www.nngroup.com/articles/utility-navigation/

Ferre, X., Juristo, N., Windl, H., & Constantine, L. (2001). Usability basics for software developers. *IEEE Software*, *18*(1), 22–29.

Field, A. P. (2009). *Discovering statistics using SPSS* (3rd ed.). SAGE Publications.

Fogg, B. J., Marshall, J., Laraki, O., Osipovich, A., Varma, C., Fang, N., … Hall, C. (2001). What Makes Web Sites Credible ? A Report on a Large Quantitative Study. In *SIGCHI conference on Human factors in computing systems* (pp. 61–68).

Fox, D., Sillito, J., & Maurer, F. (2008). Agile Methods and User-Centered Design: How These Two Methodologies are Being Successfully Integrated in Industry. In *Agile 2008 Conference* (pp. 63–72). IEEE.

Frøkjær, E., & Hornbæk, K. (2008). Metaphors of human thinking for usability inspection and design. *ACM Transactions on Computer-Human Interaction*, *14*(4), 1–33.

Galitz, W. O. (2007). *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques* (3rd ed.). John Wiley & Sons.

Gathercole, S. E. (1995). Is nonword repetition a test of phonological memory or long-term knowledge? It all depends on the nonwords. *Memory & Cognition*, *23*(1), 83–94.

Gill, P., Stewart, K., Treasure, E., & Chadwick, B. (2008). Methods of data collection in qualitative research: interviews and focus groups. *BDJ*, *204*(6), 291–295.

Glover, D., & Miller, D. (2003). Players in the management of change: Introducing interactive whiteboards into schools. *Management in Education*, *17*(1), 20–23.

Gong, C. (2009). Human-computer interaction: The usability test methods and design principles in the human-computer interface design. In *2009 2nd IEEE International Conference on Computer Science and Information Technology* (pp. 283–285). IEEE.

Gould, J. D., Boies, S., & Ukelson, J. (1997). How to Design Usable Systems. In *Handbook of Human Computer Imteraction* (2nd ed., pp. 231–254). Elsevier B.V.

Gould, J. D., & Lewis, C. (1985). Designing for Usability: Key Principles and What Designers Think. *Communications of the ACM*, *28*(3), 300–311.

Gould, S., & Powell, P. (2004). Understanding Organisational Knowledge. *Journal of Decision Systems*, *13*(2), 183–202.

Grix, J. (2002). Introducing students to the generic terminology of social research. *Politics*, *22*(3), 175–186.

Gulliksen, J., & Göransson, B. (2001). Reengineering the System Development Process for User Centred Design. In *NTERACT* (pp. 359–366).

Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J., & Cajander, Å. (2003). Key principles for user-centred systems design. *Behaviour and Information Technology*, *22*(6), 397–409.

Gutwin, C., & Greenberg, S. (2000). The mechanics of collaboration: developing low cost usability evaluation methods for shared workspaces. In *Proceedings IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE 2000)* (pp. 98–103). IEEE Comput. Soc.

Halabi, L. (2007). Expert usability review vs. usability testing. Retrieved from http://www.webcredible.com/blog/expert-usability-review-vs-usability-testing

Halverson, T., & Hornof, A. J. (2004). Link colors guide a search. In *Extended abstracts of the 2004 conference on Human factors and computing systems - CHI '04* (p. 1367). New York, New York, USA: ACM Press.

Han, S. H., Yun, M. H., Kwahk, J., & Hong, S. W. (2001). Usability of consumer electronic products. *International Journal of Industrial Ergonomics*, *28*(3–4), 143–151.

Hasan, L. (2009). *Usability evaluation framework for e-commerce websites in developing countries*. Loughborough University. Retrieved from https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/5647

Henderson, R., Podd, J., Smith, M., & Varela-Alvarez, H. (1995). An examination of four user-based software evaluation methods. *Interacting with Computers*, *7*(4), 412–432.

Hix, D., & Hartson, H. R. (1993). *Developing user interfaces : ensuring usability through product & process*. J. Wiley.

Holloway, I., & Todres, L. (2003). The Status of Method: Flexibility, Consistency and Coherence. *Qualitative Research*, *3*(3), 345–357.

Holzinger, A. (2005). Usability engineering methods for software developers. *Communications of the ACM*, *48*(1), 71–74.

Howarth, J., Smith-Jackson, T., & Hartson, R. (2009). Supporting novice usability practitioners with usability engineering tools. *International Journal of Human-Computer Studies*, *67*(6), 533–549.

Huizingh, E. K. R. . (2000). The content and design of web sites: an empirical study. *Information & Management*, *37*(3), 123–134.

Human-memory. (2010a). Long-Term Memory. Retrieved January 25, 2016, from http://www.human-memory.net/types_long.html

Human-memory. (2010b). Short-Term Memory and Working Memory. Retrieved January 22, 2016, from http://www.human-memory.net/types_short.html

Human-memory.net. (2010). Types of Memory - The Human Memory. Retrieved October 12, 2015, from http://www.human-memory.net/types.html

Humayoun, S. R. (2011). *Incorporating Usability Evaluation in Software Development Environments*. "Sapienza" Universit`a di Roma.

Humayoun, S. R., Dubinsky, Y., & Catarci, T. (2011). A Three-Fold Integration Framework to Incorporate User–Centered Design into Agile Software Development. In *Human centered design* (pp. 55–64).

Hussain, Z., Slany, W., & Holzinger, A. (2009). Current state of agile user-centered design: A survey. In *HCI and Usability for e-Inclusion* (pp. 416–427). Springer Berlin Heidelberg.

Hussain, Z., Slany, W., & Holzinger, A. (2009). Investigating Agile User-Centered Design in Practice: A Grounded Theory Perspective. In *HCI and Usability for e-Inclusion* (Vol. 5889, pp. 279–289).

Hwang, W., & Salvendy, G. (2010). Number of People Required for Usability Evaluation: The 10±2 Rule. *Communications of the ACM*, *53*(5), 130–133.

Ishii, H. (2008). The tangible user interface and its evolution. *Communications of the ACM*, *51*(6), 32–36.

ISO. (2000). *Information technology — Software product quality*. *Iso/Iec Fdis 9126-1* (Vol. 2000).

ISO 13407. (1999). *Human-centred design processes for interactive systems*. The international organization for standardization.

ISO 9241-210. (2010). *Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems*. *iso.org*. The international organization for standardization.

Ivory, M. Y., & Chevalier, A. (2002). *A study of automated web site evaluation tools*. *University of Washington, Department of Computer …*.

Ivory, M. Y., & Hearst, M. a. (2001). The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys*, *33*(4), 470–516.

Jacobsen, N. E. (1999). *Usability Evaluation Methods The Reliability and Usage of Cognitive Walkthrough and Usability Test*. University of Copenhagen.

Jain, J., Lund, A., & Wixon, D. (2011). The future of natural user interfaces. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11* (pp. 211–214). New York, New York, USA: ACM Press.

James, M. (2010). Scrum Reference Card Scrum Meetings. Retrieved from http://scrumreferencecard.com/

Johnson, R. B., & Onwuegbuzie, A. J. (2004). Mixed Methods Research: A Research Paradigm Whose Time Has Come. *Educational Researcher*, *33*(7), 14–26.

Johnson, T. (2013). Evaluating the Usability of Collapsible Sections (or jQuery's Content Toggle). Retrieved October 23, 2015, from http://idratherbewriting.com/2013/03/25/evaluating-the-usability-of-collapsible-sections-or-jquerys-content-toggle/#comment-2029521352

Jokela, T., Iivari, N., Matero, J., & Karukka, M. (2003). The Standard of User-Centered Design and the Standard Definition of Usability : Analyzing ISO 13407 against ISO. In *In Proceedings of the Latin American conference on Human-computer interaction* (pp. 53–60). ACM.

Jorgensen, M., & Papatheocharous, E. (2015). Believing is Seeing: Confirmation Bias Studies in Software Engineering. In *2015 41st Euromicro Conference on Software Engineering and Advanced Applications* (pp. 92–95). IEEE.

Juristo, N., Moreno, A. M., & Sanchez-Segura, M.-I. (2007). Analysing the impact of usability on software design. *Journal of Systems and Software*, *80*(9), 1506–1516.

Kane, D. (2003). Finding a place for discount usability engineering in agile development:

throwing down the gauntlet. In *Proceedings of the Agile Development Conference, 2003. ADC 2003* (pp. 40–46). IEEE.

Karat, C.-M. (1997). Cost-justifying usability engineering in the software life cycle. In M. G. Helander, T. K. Landauer, & P. V Prabhu (Eds.), *Handbook of Human-Computer Interaction* (p. 1582). Elsevier.

Katz, R. (1999). *Dancing with the Devil: Information Technology and the New Competition in Higher Education. Jossey-Bass Higher and Adult Education Series*. ossey-Bass Publishers, 350 Sansome St., San Francisco, CA 94104.

Keinonen, T. (2010). Protect and appreciate – Notes on the justification of user-centered design. *International Journal of Design*, *4*(1), 17–27.

Kheterpal, S. (2002). Usability On The Cheap. Retrieved July 16, 2015, from http://www.sitepoint.com/usability-cheap/

King, N., & Horrocks, C. (2010). *Interviews in qualitative research*. SAGE.

Kitzinger, J. (1995). Qualitative research. Introducing focus groups. *BMJ (Clinical Research Ed.)*, *311*(7000), 299–302.

Kjeldskov, J., Skov, M. B., Als, B. S., & Høegh, R. T. (2004). Is it worth the hassle? Exploring the added value of evaluating the usability of context-aware mobile systems in the field. *Mobile Human-Computer Interaction - MobileHCI*, 61–73.

Koehler, M., Mishra, P., Kereluik, K., & Shin, T. (2014). The technological pedagogical content knowledge framework. *Handbook of Research on Educational Communications and Technology*, 101–111.

Kräenbring, J., Monzon Penza, T., Gutmann, J., Muehlich, S., Zolk, O., Wojnowski, L., … Sarikas, A. (2014). Accuracy and completeness of drug information in Wikipedia: a comparison with standard textbooks of pharmacology. *PloS One*, *9*(9), e106930.

Kujala, S. (2003). User involvement: A review of the benefits and challenges. *Behaviour & Information Technology*, *22*(1), 1–16.

Lewis, J. R., & Sauro, J. (2009). The factor structure of the system usability scale. In *Human Centered Design* (pp. 94–103). Springer Berlin Heidelberg.

Liu, F., Zuo, M., & Zhang, P. (2011). Human-Machine Function Allocation In Information Systems: A Comprehensive Approach. In *PACIS 2011 Proceedings* (p. 117).

Lutz, M., Boucher, X., & Roustant, O. (2013). Methods and applications for IT capacity decisions: Bringing management frameworks into practice. *Journal of Decision Systems*, *22*(4), 332–355.

Magazine, S. (2009). Breadcrumbs In Web Design: Examples And Best Practices. Retrieved

October 23, 2015, from http://www.smashingmagazine.com/2009/03/breadcrumbs-in-web-design-examples-and-best-practices/

Maguire, M. (2001). Methods to support human-centred design. *International Journal of Human-Computer Studies*, *55*(4), 587–634.

Maguire, M. C. (1999). A review of user-interface design guidelines for public information kiosk systems. *International Journal of Human Computer Studies*, *50*(3), 263–280.

Mandel, T. (1997). *The Elements of User Interface Design:* (1st ed.). John Wiley & Sons.

Marc, M. (2013). 7 Benefits of Agile and User Centered Design. Retrieved September 17, 2015, from https://www.thoughtworks.com/insights/blog/agile-and-user-centered-design

Maroney, D. (1997). In praise of hypertext. *Journal of Advertising Research*, *37*(2), 7–9.

Marshall, C., & Gretchen B, R. (2006). Data Collection Methods. In *Designing Qualitative Research* (4th ed., p. 262). Sage Publications.

Marshall, G. (2005). The purpose, design and administration of a questionnaire for data collection. *Radiography*, *11*(2), 131–136.

Mayer, R. E. (2009). *Multimedia Learning* (2nd ed.). Cambridge University Press.

Mayer, R. E., & Moreno, R. (2010). Nine Ways to Reduce Cognitive Load in Multimedia Learning Nine Ways to Reduce Cognitive Load in Multimedia Learning. *Educational Psychologist*, *1520*(38), 43–52.

Mohammed, N., Munassar, A., Govardhan, A., & Pradesh, A. (2010). A Comparison Between Five Models Of Software Engineering. *IJCSI International Journal of Computer Science Issues*, *7*(5), 94–101.

Mohd Sam, M. F., & Tahir, M. N. H. (2009). Website Quality and Consumer Online Purchase Intention of Air Ticket. *International Journal of Basic & Applied Sciences*, *9*(10), 20–25.

Moore, N. (2006). *How to do research : a practical guide to designing and managing research projects* (3rd ed.). Facet Publishing.

Myers, G., Sandler, C., & Badgett, T. (2011). *The art of software testing*. John Wiley & Sons.

Myers, M. (1997). Qualitative research in information systems. *Management Information Systems Quarterly*, *21*(2), 241–242.

Najafi, M., & Toyoshiba, L. (2008). Two Case Studies of User Experience Design and Agile Development. In *Agile 2008 Conference* (pp. 531–536). IEEE.

NASA Administrator. (2014, June 5). Tools. Retrieved November 14, 2015, from https://www.nasa.gov/accessibility/resources/res_tools.html#.Vkaoj6TtlBc

Nielsen, J. (1993a). *Usability Engineering*. Morgan Kaufmann.

Nielsen, J. (1993b). *Usability Engineering* (1st ed.). United States: Morgan Kaufmann.

Nielsen, J. (1994). Usability laboratories. *Behaviour & Information Technology*, *13*(1–2), 3–8. Retrieved from http://www.tandfonline.com/doi/abs/10.1080/01449299408914577

Nielsen, J. (1995). 10 Heuristics for User Interface Design. Retrieved July 16, 2015, from http://www.nngroup.com/articles/ten-usability-heuristics/

Nielsen, J. (1995). Applying discount usability engineering. *IEEE Software*, *12*(1), 98–100.

Nielsen, J. (1995). How to Conduct a Heuristic Evaluation. Retrieved from https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/

Nielsen, J. (1995). Severity Ratings for Usability Problems. *Nielsen Norman Group*, 1. Retrieved from http://www.nngroup.com/articles/how-to-rate-the-severity-of-usability-problems/

Nielsen, J. (2000). Why You Only Need to Test with 5 Users. Retrieved December 22, 2015, from https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

Nielsen, J. (2006). Quantitative Studies: How Many Users to Test? Retrieved February 11, 2016, from https://www.nngroup.com/articles/quantitative-studies-how-many-users/

Nielsen, J. (2007). Fast, Cheap, and Good: Yes, You Can Have It All. Retrieved December 2, 2015, from http://www.nngroup.com/articles/fast-cheap-and-good-methods/

Nielsen, J. (2008). Top 10 Application-Design Mistakes. Retrieved from https://www.nngroup.com/articles/top-10-application-design-mistakes/

Nielsen, J. (2012). Usability 101: Introduction to Usability. Retrieved August 18, 2016, from https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Nielsen, J., & Mack, R. L. (1994). *Usability Inspection Methods* (1st ed.). United States: John Wiley & Sons Inc.

Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems Empowering people - CHI '90* (pp. 249–256). New York, New York, USA: ACM Press.

Nielsen, J., & Tahir, M. (2001). *Homepage Usability: 50 Websites Deconstructed* (1st ed.). New Riders.

Norman, D. A. (2013). *The Design of Everyday Things: Revised and Expanded Edition*. Basic Books.

Nutschan, C. (2008). Spiral model (Boehm, 1988). Retrieved October 18, 2015, from https://commons.wikimedia.org/wiki/File:Spiral_model_(Boehm,_1988).png

Odgaard-Jensen, J., Vist, G., & Timmer, A. (2011). Randomisation to protect against

selection bias in healthcare trials. *The Cochrane Library*. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002/14651858.MR000012.pub3/full

Pajares, F., & Schunk, D. (2001). The development of academic self-efficacy. *Development of Achievement Motivation*.

Patrick W, J. (1998). *An Introduction to Usability*. CRC Press.

Patton, M. Q. (2002). *Qualitative research and evaluation methods*. Sage Publications.

Perry, W. E. (2006). *Effective methods for software testing*. Wiley.

Petrie, H., & Bevan, N. (2009). The evaluation of accessibility, usability and user experience. In *The universal access handbook* (pp. 10–20).

Peytchev, A., Couper, M. P., McCabe, S. E., & Crawford, S. D. (2006). Web Survey Design: Paging versus Scrolling. *Public Opinion Quarterly*, *70*(4), 596–607.

Preece, J., Benyon, D., & University, O. (1993). *A Guide to Usability: Human Factors in Computing*. Addison-Wesley Longman Publishing Co., Inc.

Preece, J., Rogers, Y., & Sharp, H. (2002). *Interaction Design: Beyond Human-Computer Interaction* (1st ed.). United States: John Wiley &; Sons Ltd.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., & Carey, T. (1994). *Human Computer Interaction* (1st ed.). Addison Wesley.

Pressman, R. S. (2010). *Software Engineering A Practitioner's Approach* (7th ed.). McGraw-Hill Education.

Rajanen, M. (2003). Usability cost-benefit models–different approaches to usability benefit analysis. In *26th Information Systems Research Seminar In Scandinavia (IRIS26)*.

Ramli, R. bt M., & Jaafar, A. bt. (2008). e-RUE : A cheap possible solution for usability evaluation. In *2008 International Symposium on Information Technology* (Vol. 3, pp. 1–5). IEEE.

Rannikko, P. (2011). *User-Centered Design in Agile Software Development*. University of Tampere.

Remenyi, D., & Money, A. H. (2012). *Research supervision for supervisors and their students* (2nd ed.). Academic Conferences International.

Riihiaho, S., Nieminen, M., Westman, S., Addams-Moring, R., & Katainen, J. (2015). Procuring Usability: Experiences of Usability Testing in Tender Evaluation. In *Nordic Contributions in IS Research* (Vol. 223, pp. 108–120). Springer International Publishing.

Robson, C. (1993). *Real World Research : A Resource for Social Scientists and Practitioner-researchers*. Oxford: BLACKWELL.

Roehm, T., Tiarks, R., Koschke, R., & Maalej, W. (2012). How do professional developers comprehend software? In *Proceedings of the 34th International Conference on Software Engineering* (pp. 255–265). IEEE.

Ronjeffries.com. (2011). What is Extreme Programming? Retrieved September 25, 2015, from http://ronjeffries.com/xprog/what-is-extreme-programming/

Rosen, D. E., & Purinton, E. (2004). Website design: Viewing the web as a cognitive landscape. *Journal of Business Research*, *57*(7), 787–794.

Rosenbaum, S., Rohn, J. A., & Humburg, J. (2000). A toolkit for strategic usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '00* (pp. 337–344). New York, New York, USA: ACM Press.

Rouse, M. (2007). What is agile software development (ASD). Retrieved September 4, 2015, from http://searchsoftwarequality.techtarget.com/definition/agile-software-development

Rubin, J., Chisnell, D., & Spool, J. (2008). *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests* (2nd ed.). United States: Wiley, John & Sons.

Ruchika, S. (2012). A comparative study of software development models. *International Journal of Advances in Computing and Information Technology*, *1*(3), 256–274.

Ryan, G. W., & Bernard, H. R. (2000). Data Management and Analysis Methods. In *Handbook of Qualitative Research* (2nd ed.). SAGE Publications.

Salah, D. (2011). A framework for the integration of user centered design and agile software development processes. In *Proceeding of the 33rd international conference on Software engineering - ICSE '11* (p. 1132). New York, New York, USA: ACM Press.

Salah, D., Paige, R. F., & Cairns, P. (2014). A systematic literature review for agile development processes and user centred design integration. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14* (pp. 1–10). New York, New York, USA: ACM Press.

Sandelowski, M. (2000). Focus on research methods combining qualitative and quantitative sampling, data collection, and analysis techniques. *Research in Nursing & Health*.

Saul, M. L. (2008). Likert Scale. Retrieved July 14, 2015, from http://www.simplypsychology.org/likert-scale.html

Saunders, M. N. K., Lewis, P., & Thornhill, A. (2009). *Research methods for business students*. Prentice Hall.

Sauro, J. (2011a). *A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices*. CreateSpace Independent Publishing Platform.

Sauro, J. (2011b). Measuring Usability with the System Usability Scale (SUS). Retrieved

July 8, 2015, from http://www.measuringu.com/sus.php

Sauro, J. (2013). Rating the Severity of Usability Problems. Retrieved August 25, 2015, from http://www.measuringu.com/blog/rating-severity.php

Schwaber, K., & Sutherland, J. (2013). *The scrum guide: the Definitive Guide to Scrum: the rules of the Game*. *Scrum. org* (Vol. 2). Retrieved from http://www.scrum.org/scrumguides/%5Cnhttp://pdf4420.psxbook.com/scrum_1868546.pdf

Scrum. (2015). The home of Scrum. Retrieved September 8, 2015, from https://www.scrum.org/About

Scrum.org. (2015). What is Scrum? Retrieved September 28, 2015, from https://www.scrum.org/resources/what-is-scrum

Seema, & Malhotra, S. (2012). Analysis and tabular comparison of popular SDLC models. *International Journal of Advances in Computing and Information*, *1*(3), 277–286.

Shackel, B. (2009a). Human–computer interaction – Whence and whither? *Interacting with Computers*, *21*(5–6), 353–366.

Shackel, B. (2009b). Usability - Context, framework, definition, design and evaluation. *Interacting with Computers*, *21*(5–6), 339–346.

Sharp, H., Robinson, H., & Segal, J. (2004). INTEGRATING USER-CENTRED DESIGN AND SOFTWARE ENGINEERING: A ROLE FOR EXTREME PROGRAMMING? In *BCS-HCI Group's 7th Educators Workshop: Effective Teaching and Training in HC* (pp. 1–4).

Shneiderman, B. (1991). A taxonomy and rule base for the selection of interaction styles. In *Human factors for informatics usability* (pp. 325–342).

Shneiderman, B. (2015). The Eight Golden Rules of Interface Design. Retrieved December 4, 2015, from https://www.cs.umd.edu/users/ben/goldenrules.html

Shneiderman, B., Plaisant, C., Cohen, M., & Jacobs, S. (2013). *Designing the User Interface: Pearson New International Edition: Strategies for Effective Human-Computer Interaction*. United Kingdom: Pearson Education Limited.

Shone, A. (2010). Designing a kanban board – not as simple as you might think. Retrieved October 30, 2015, from http://blog.caplin.com/2010/02/16/designing-a-kanban-board-not-as-simple-as-you-might-think/

Silva da Silva, T., Martin, A., Maurer, F., & Silveira, M. (2011). User-Centered Design and Agile Methods: A Systematic Review. In *2011 AGILE Conference* (pp. 77–86). IEEE.

Silva da Silva, T., Selbach Silveira, M., Maurer, F., & Hellmann, T. (2012). User Experience

Design and Agile Development: From Theory to Practice. *Journal of Software Engineering and Applications*, *5*(10), 743–751.

Silva da Silva, T., Silveira, M. S., & Maurer, F. (2015). Usability Evaluation Practices within Agile Development. In *48th Hawaii International Conference on System Sciences* (pp. 5133–5142). IEEE.

Skov, M. ., & Stage, J. (2012). Training Software Developers and Designers to Conduct Usability Evaluations. *Behaviour & Information Technology*, *31*(4), 425–435.

Skov, M. B., & Stage, J. (2005). Supporting problem identification in usability evaluations. *Proceedings of the 17th Australia Conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future. Computer-Human Interaction Special Interest Group (CHISIG) of Australia*, 1–9.

Sohaib, O., & Khan, K. (2010). Integrating usability engineering and agile software development: A literature review. In *2010 International Conference On Computer Design and Applications* (Vol. 2, pp. V2-32-V2-38). IEEE.

Sohaib, O., & Khan, K. (2011). Incorporating discount usability in extreme programming. *International Journal of Software Engineering and Its Applications*, *5*(1), 51–62.

Sommerville, I. (2010). *Software Engineering* (9th ed.). Pearson Education.

Song, J. H., & Zinkhan, G. M. (2003). FEATURES OF WEB SITE DESIGN, PERCEPTIONS OF WEB SITE QUALITY, AND PATRONAGE BEHAVIOR. In *ACME Proceedings* (pp. 106–114).

Song, L., Singleton, E. S., Hill, J. R., & Koh, M. H. (2004). Improving online learning: Student perceptions of useful and challenging characteristics. *The Internet and Higher Education*, *7*(1), 59–70.

Sparrow, P. (2012). Spiral Model : Advantages and Disadvantages. Retrieved August 13, 2015, from http://www.ianswer4u.com/2011/12/spiral-model-advantages-and.html#axzz3ihy97bFN

Stenius, K., Mäkelä, K., & Miovsky, M. (2008). How to write publishable qualitative research. *Addiction Science: A Guide for the Perplexed,UK: International Society of Addiction Journal Editors*, 82–97. Retrieved from http://www.inclentrust.org/inclen/uploadedbyfck/file/compile resourse/Qualitative Research/How to publish qualitative research.pdf

Stone, D., Woodroffe, M., Minocha, S., & Jarrett, C. (2005). *User Interface Design and Evaluation*. United States: Morgan Kaufmann.

Sutcliffe, A. (1988). *Human-Computer Interface Design* (1st ed.). London: Macmillan

Education UK.

Tang, K., & Davis, A. (1995). Critical factors in the determination of focus group size. *Family Practice*, *12*(4), 474–475.

Tashakkori, A., & Teddlie, C. (2008). Quality of inferences in mixed methods research: Calling for an integrative framework. *Advances in Mixed Methods Research*, 101–119.

Teoh, C. (2006). User-centred design (UCD) - 6 methods -. Retrieved September 2, 2015, from http://www.webcredible.com/blog/user-centered-design-ucd-6-methods/

Thatcher, J. (2011a). Evaluation of Bobby. Retrieved August 17, 2015, from http://jimthatcher.com/bobbyeval.htm

Thatcher, J. (2011b). LIFT from UsableNet. Retrieved November 14, 2015, from http://jimthatcher.com/lifteval.htm

Thermistocleous, M. (2002). *Evaluating the adoption of enterprise application integration in multinational organisations*. Brunel University. Retrieved from http://dl.merc.ac.ir/handle/Hannan/27799

Tomei, L. A. (2010). Learning Theories and Pedagogy: Teaching the Traditional Learner. In *Designing Instruction for the Traditional, Adult, and Distance Learner: A New Engine for Technology-Based Teaching* (pp. 1–14). IGI Global.

Travis, D. (2009). How to prioritise usability problems. Retrieved August 25, 2015, from http://www.userfocus.co.uk/articles/prioritise.html

Travis, D. (2011). ISO 13407 is dead. Long live ISO 9241-210! Retrieved October 27, 2015, from http://www.userfocus.co.uk/articles/iso-13407-is-dead.html

Usability.gov. (, October). Scenarios. Retrieved October 1, 2015, from http://www.usability.gov/how-to-and-tools/methods/scenarios.html

Usability.gov. (2013a). Reporting Usability Test Results.

Usability.gov. (2013b, September 6). System Usability Scale (SUS). Retrieved August 4, 2015, from http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

Usability.gov. (2013c, November 13). Usability Testing. Retrieved August 24, 2015, from http://www.usability.gov/how-to-and-tools/methods/usability-testing.html

Usability First. (2015). Ten minute rule. Retrieved September 21, 2015, from http://www.usabilityfirst.com/glossary/ten-minute-rule/

van Velsen, L., van der Geest, T., Klaassen, R., & Steehouder, M. (2008). User-centered evaluation of adaptive and adaptable systems: a literature review. *The Knowledge Engineering Review*, *23*(3), 261–281.

Venkataramani, K. (2014). Analysis of Software Process Models and Applications. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, *3*(3), 203–207.

VersiononeSurvey. (2008). *The State of Agile Development*.

Vertegaal, R., & Poupyrev, I. (2008). Organic user interfaces. *Communications of the ACM*, *51*(6), 30–36.

Vredenburg, K., Smith, P. W., Carey, T., & Mao, J.-Y. (2002). A survey of user-centered design practice. In *Proceedings of theSIGCHI conference on Human factors in computing systems Changing our world, changing ourselves - CHI '02* (Vol. 4, pp. 471–478).

Wall-Skills.com. (2014). Scrum Events. Retrieved September 28, 2015, from http://wall-skills.com/2014/scrum-events/

Walliman, N. (2006). *Social research methods*. SAGE.

Ward, C., & Benson, S. (2010). Developing new schemas for online teaching and learning: TPACK. *Journal of Online Learning and Teaching*, *6*(2).

Watchfire. (2005). Watchfire Bobby 5.3 User Guide. Retrieved November 14, 2015, from http://molar.crb.ucp.pt/cursos/P%25C3%25B3s-Gradua%25C3%25A7%25C3%25B5es/EE-2007/Sess%25C3%25A3o 19 Janeiro/Bobby - User Guide.pdf

WAVE Web Accessibility evaluation tool. (n.d.). WAVE Help. Retrieved November 14, 2015, from http://wave.webaim.org/help

Wells, D. (2013). Extreme Programming. Retrieved October 25, 2015, from https://commons.wikimedia.org/wiki/File:Extreme_Programming.svg

Whittaker, J. A. (2000). What is software testing? And why is it so hard? *IEEE Software*, *17*(1), 70–79.

Wiegers, K., & Beatty, J. (2013). *Software requirements*. pe. Retrieved from https://books.google.co.uk/books?hl=en&lr=&id=nbpCAwAAQBAJ&oi=fnd&pg=PT32&dq=collect+the+software+requirements&ots=9nQWC_2tVl&sig=9T0bNBg_YUyyAMwjLcDCNjFTDtg

Wikipedia. (2015). Retinal scan. Retrieved October 12, 2015, from https://en.wikipedia.org/wiki/Retinal_scan

Williams, D. (2007). Reactable Multitouch. Retrieved October 11, 2015, from https://commons.wikimedia.org/wiki/File:Reactable_Multitouch.jpg

Yang, X., & Chen, G. (2009). Human-Computer Interaction Design in Product Design. In

2009 First International Workshop on Education Technology and Computer Science (Vol. 2, pp. 437–439). IEEE. http://doi.org/10.1109/ETCS.2009.359

Yin, R. K. (1994a). *Case study research : design and methods*. Sage Publications.

Yin, R. K. (1994b). *Case study research: design and methods* (2nd ed.). Sage Publications.

Yin, R. K. (2003). *Case study research : design and methods* (3rd ed.). SAGE Publications.

Yin, R. K. (2013). *Case study research : design and methods* (5th ed.). SAGE Publications.